ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ - WEBINTELLIGENCE

Μελέτη και χρήση της πλατφόρμας ASP.NET για την ανάπτυξη διαδικτυακών εφαρμογών

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΛΟΥΦΗ ΙΝΑ ΚΩΝΣΤΑΝΤΙΝΑ

Επιβλέπων : Μιχαήλ Σαλαμπάσης
Καθηγητής,Α.Τ.Ε.Ι.Θ

Θεσσαλονίκη, Μάϊος 2017

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ - WEBINTELLIGENCE

Μελέτη και χρήση της πλατφόρμας ASP.NET για την ανάπτυξη διαδικτυακών εφαρμογών

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΙΝΑ ΚΩΝΣΤΑΝΤΙΝΑ ΛΟΥΦΗ

Επιβλέπων : Μιχαήλ Σαλαμπάσης
Καθηγητής,Α.Τ.Ε.Ι.Θ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 21 Ιουνίου 2017.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Μιχαήλ Σαλαμπάσης	Ευκλείδης Κεραμόπουλος	Ευστάθιος Αντωνίου
ΚαθηγητήςΑ.Τ.Ε.Ι.Θ.	ΚαθηγητήςΑ.Τ.Ε.Ι.Θ.	ΚαθηγητήςΑ.Τ.Ε.Ι.Θ.

Θεσσαλονίκη, Μάϊος 2017

Υπόδειγμα φύλλου τίτλου (πίσω σελίδα) του αντιτύπου που υποβάλλεται στις βιβλιοθήκες (διπλωματική εργασία)

(Υπογραφή)

.....

ΙΝΑ ΚΩΝΣΤΑΝΤΙΝΑ ΛΟΥΦΗ

Μηχανικός Πληροφορικής ΑΤΕΙ Δυτικής Μακεδονίας

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή της διπλωματικής μου εργασίας κύριο Μιχάλη Σαλαμπάση για την άριστη συνεργασία και την πολύτιμη βοήθεια που μου προσέφερε σε όλη την διάρκεια της έρευνας και της υλοποίησης της εργασίας αυτής.

Επίσης, ένα μεγάλο ευχαριστώ στην οικογένειά μου, για συνεχή στήριξη, οικονομική και ηθική, καθ' όλη την διάρκεια των σπουδών μου.

Περίληψη

Ο σκοπός της διπλωματικής εργασίας είναι να μελετήσει,να παρουσιάσει και να συγκρίνει τα διάφορα μοντέλα ανάπτυξης διαδικτυακών εφαρμογών με χρήση της πλατφόρμαςASP.NET. Συγκεκριμένα θα μελετηθούν μοντέλα όπως Web Forms, Web Pages/Razor και Single Page Apps. Θα παρουσιαστούν επίσης μεθοδικά τα εργαλεία που είναι διαθέσιμα για την ανάπτυξη εφαρμογών με αυτά τα μοντέλα. Τέλος, θα γίνει μια συγκριτική αξιολόγηση αυτών των διαφορετικών μοντέλων.

Συγκεκριμένα, στο κεφάλαιο 1 θα γίνει μια εισαγωγή αναφέροντας τις διαφορές των διαδικτυακών εφαρμογών με τις desktop εφαρμογές, τις τυπικές αρχιτεκτονικές που υπάρχουν, τις τεχνολογίες client/server sideκαι τον κύκλο ανάπτυξης μιας διαδικτυακής εφαρμογής.

Στο επόμενο κεφάλαιο (2) θα μελετήσουμε κάποια εργαλεία που είναι διαθέσιμα για την ανάπτυξη web εφαρμογών καθώς και τα Συστήματα Διαχείρισης Περιεχομένου (CMS).

Στο κεφάλαιο 3 πλέον θα παρουσιάσουμε τα συστατικά στοιχεία της ASP.NET, την αρχιτεκτονική της, τα μοντέλα των εφαρμογών και τις αρχές σχεδιασμού που διέπουν την ASP.NET.

Στο βασικό μας κεφάλαιο πλέον, στο κεφάλαιο 4, θα μελετήσουμε τα μοντέλα ανάπτυξης διαδικτυακών εφαρμογών με την χρήση της ASP.NET. Θα ξεκινήσουμε με τις Web Forms και τα χαρακτηριστικά τους. Θα συνεχίσουμε με τις Master Pages και τα ASP.NET Ajax Controls. Παρακάτω, θα εστιάσουμε στις προκλήσεις των Web Forms και στις διαφορές τους με το MVC.

Στα κεφάλαια 5 και 6 θα γίνει συζήτηση για τις Web Pages και την ASP.NET Razor αντίστοιχα.

Στο επόμενο κεφάλαιο, παρουσιάζουμε τα Single Page Apps και τη δημιουργία ενός μικρού project σε Visual Studio. Κλείνοντας το κεφάλαιο 7, θα γίνει μια σύγκριση/αναφορά στα μοντέλα MVC/MVVM.

Στο τελευταίο μας κεφάλαιο, το κεφάλαιο 8, θα προσπαθήσουμε να κάνουμε μια ποιοτική αξιολόγηση των μοντέλων που αναλύθηκαν προηγουμένως και θα συγκεντρωθούν κάποια συμπεράσματα.

Λέξεις Κλειδιά:<<ASP.NET, Master Pages, Web Forms, MVC,CMS, Single Page Apps>>

Η σελίδα αυτή είναι σκόπιμα λευκή.

Abstract

The scope of this thesis is to study and to present the various models of development web applications usind the ASP.NET platform.Specifically, will study models as Web Forms, Web Pages/Razor and Single Page Apps.Furthermore, will present methodically the tools which are available for the development of applications with these models. Finally, will be a comparative assessment of the models.

In more detail, at the first chapter will be an introduction to web applications, to their differences from desktop applications, to the 3-tier architecture where existed. Moreover, will be studied the server side and client side technologies and the cycle development of a web application.

At the next chapter will study some tools which are available for web applications development and the Content Management Systems (CMS).

At the third chapter will present the ingredients of ASP.NET, the architecture of ASP.NET, the models of the applications and the design's authorities that govers the ASP.NET.

At our main chapter now, the forth chapter, will peruse with details the models of web applications development using the ASP.NET platform. We will start analyzing the Web Forms and their feautures.After that will continue the research with Master Pages and the ASP.NET Ajax Controls. Below will focuse to the challenges of the Web Forms and the differences between Web Forms and MVC.

At chapters 5 and 6 will be a study for Web Pages and for ASP.NET Razor.

The main theme at the next chapter are the Single Page Apps and the creation of a small project written in Visual Studio. Closing the chapter 7 will be a comparison between the MVC model an MVVM model.

At the last chapter, in chapter 8, will be a qualitative evaluation of the models which are analyzed previously and will be collected any conclusions.

Keywords:<<ASP.NET, Master Pages, Web Forms, MVC,CMS, Single Page Apps>>

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Κεφάλαιο1
Ανάπτυξη διαδικτυακών συστημάτων & εφαρμογών

Στο κεφάλαιο 1, πρόκειται να δούμε τα βασικά στοιχεία των διαδικτυακών εφαρμογών, καθώς και τις διαφορές που έχουν με τις desktop εφαρμογές. Επίσης θα μελετήσουμε τα web application frameworks. Στη συνέχεια θα παραθέσουμε τα πλεονεκτήματα αλλά και τα μειονεκτήματα των desktop εφαρμογών και των framework για ανάπτυξη web εφαρμογών.Αμέσως μετά θα μιλήσουμε για για τις τυπικές αρχιτεκτονικές (3-tier), το μοντέλο client-server, το ρόλο πελάτη και εξυπηρετητή, τις τεχνολογίες server side, τη σχεδίαση και τον κύκλο ζωής των εφαρμογών, τα μοντέλα του κύκλου ζωής, τη διαδικασία ανάπτυξης καθώς και τον έλεγχο των web εφαρμογών.

1.1 Διαδικτυακές εφαρμογές

Οι διαδικτυακές εφαρμογές είναι δυναμικές ιστοσελίδες συνδυασμένες με server-side προγραμματισμό που παρέχουν διάφορες λειτουργίες όπως η αλληλεπίδραση με τους χρήστες, η σύνδεση με back-end βάσεις δεδομένων και η δημιουργία αποτελεσμάτων για τα προγράμματα περιήγησης.

Παραδείγματα διαδικτυακών εφαρμογών είναι το online banking, η κοινωνική δικτύωση, οι ηλεκτρονικές κρατήσεις, το ηλεκτρονικό εμπόριο, τα διαδραστικά παιχνίδια, η ηλεκτρονική εκπαίδευση, οι ηλεκτρονικές δημοσκοπήσεις, τα Blogs, τα ηλεκτρονικά φόρουμ, τα συστήματα διαχείρισης περιεχομένου, κ.λπ.

1.1.1 Web Application Frameworks

Τα Web Application Frameworks είναι προγραμματιστικές βιβλιοθήκες, που επιτρέπουν στους προγραμματιστές να δημιουργήσουν και να συντηρήσουν πολύπλοκα έργα διαδικτυακών εφαρμογών, χρησιμοποιώντας μια πιο γρήγορη και πιο αποτελεσματική προσέγγιση.

Τα Web Application Frameworks έχουν σχεδιαστεί για την βελτίωση του προγραμματισμού, δίνοντας έμφαση στην τεκμηρίωση και στην ευρεία χρήση βιβλιοθηκών (επαναχρησιμοποιήσιμοι κώδικες για κοινές συναρτήσεις και κλάσεις).

1.1.2 Διαφορές με desktop εφαρμογές

Η ανάπτυξη των σύγχρονων εφαρμογών λογισμικού ξεκίνησε με τις desktop εφαρμογές, οι οποίες θα μπορούσαν να χρησιμοποιηθούν μόνο σε desktop υπολογιστές. Ωστόσο, με την έλευση του διαδικτύου και του ηλεκτρονικού εμπορίου, η ανάπτυξη διαδικτυακών εφαρμογών απέκτησε μεγάλη σημασία. Οι επεξεργαστές κειμένου και οι media-players μπορούν να θεωρηθούν ως τυπικές desktop εφαρμογές, ενώ μια ιστοσελίδα ηλεκτρονικού εμπορίου μπορεί να θεωρηθεί ως μια διαδικτυακή εφαρμογή.

Εξ ορισμού, μια desktop εφαρμογή σημαίνει οποιοδήποτε λογισμικό που μπορεί να εγκατασταθεί και να εκτελεστεί πληρως σε έναν υπολογιστή (laptop ή desktop). Ορισμένες εφαρμογές desktop μπορούν επίσης να χρησιμοποιηθούν από πολλούς χρήστες σε ένα δικτυωμένο περιβάλλον. Η ανάπτυξη διαδικτυακών εφαρμογών, ωστόσο, σύντομα άρχισε να αντικαθιστά τις desktop εφαρμογές για λόγους φορητότητας και χρηστικότητας. Στους Πίνακες 1 και 2 φαίνονται τα πλεονεκτήματα και τα μειονεκτήματα (για τον προγραμματιστή) της κατασκευής desktop εφαρμογών και της κατασκευής διαδικτυακών εφαρμογών, αντίστοιχα.

Πίνακας 1. Πλεονεκτήματα και μειονεκτήματα κατασκευής desktop εφαρμογών.

ΠΛΕΟΝΕΚΤΗΜΑΤΑ
Είναι εύκολο να δημιουργηθεί μόνο ένα / .zip / .jar ή .exe αρχείο για τη διανομή τελικού προϊόντος
Η κατασκευή της γραφικής διεπαφής (GUI) είναι εύκολη τις περισσότερες φορές
Ο κώδικας είναι πιο καθαρός και δεμένος σε σχέση με τις διαδικτυακές εφαρμογές
Η ασφάλεια της εφαρμογής είναι πιο εύκολο έργο από ό, τι σε εφαρμογές web
Οι κύριες γλώσσες σε αυτό το είδος του προγραμματισμού (π.χ. C, C #, Java, VB, κτλ.) είναι σταθερές και παγιωμένες.
ΜΕΙΟΝΕΚΤΗΜΑΤΑ
Η πώληση των προγραμμάτων είναι πιο δύσκολη, ειδικά για ανεξάρτητες παραγωγές.
Υπάρχουν πολλά πράγματα που πρέπει να ξέρει ο προγραμματιστής στη γλώσσα της επιλογής του.
Οι άνθρωποι κατευθύνονται όλο και περισσότερο προς το web.

Πίνακας 2. Πλεονεκτήματα και μειονεκτήματα κατασκευής web εφαρμογών.

ΠΛΕΟΝΕΚΤΗΜΑΤΑ
Είναι ευκολότερο να δοκιμάζεται σε ευρεία κλίμακα ο κώδικας, ενώ αναπτύσσεται
Υπάρχουν διαθέσιμα πάρα πολλά εγχειρίδια για την ανάπτυξη web εφαρμογών
Το Web αρχίζει σιγά σιγά να επιταχύνεται ως επιλογή των εταιρειών για τις εφαρμογές τους
Πολλές τεχνολογίες διαθέσιμες για ανάπτυξη εφαρμογών
Οι τεχνολογίες web αναπτύσσονται συνήθως γρηγορότερα σε σχέση με τις desktop τεχνολογίες
ΜΕΙΟΝΕΚΤΗΜΑΤΑ
Πολύπλοκες, αν δεν υπάρχει σχεδιαστική γνώση
Με πολλές τεχνολογίες έρχεται η σύγχυση σχετικά με το τι πρέπει να χρησιμοποιήσετε
Η αποσφαλμάτωση στις web εφαρμογές είναι πιο δύσκολη
Η προστασία των web εφαρμογών από τους χάκερ είναι πιο δύσκολη
Αρκετός ανταγωνισμός
Όσο πιο περίπλοκη είναι η web app, τόσο πιο δύσκολο είναι να βρεθούν δωρεάν λύσεις στο Διαδίκτυο

Ακολουθεί μια βασική σύγκριση για desktop και web-based εφαρμογές που βασίζεται σε ορισμένες παραμέτρους:

- Συντήρηση – οι web-based εφαρμογές πρέπει να εγκατασταθούν μόνο μια φορά, ενώ οι desktop εφαρμογές πρέπει να εγκατασταθούν ξεχωριστά σε κάθε υπολογιστή. Επίσης, η ενημέρωση των εφαρμογών είναι χρονοβόρα στις desktop εφαρμογές, καθώς πρέπει να γίνει σε κάθε υπολογιστή, κάτι που δεν συμβαίνει με τις εφαρμογές web.
- Ευκολία στη χρήση - οι εφαρμογές desktop περιορίζονται σε μια φυσική θέση και ως εκ τούτου έχουν περιορισμό στη χρηστικότητα. Οι web εφαρμογές από την άλλη καθιστούν εύκολη για τους χρήστες την απόκτηση πρόσβασης στην εφαρμογή από οποιαδήποτε θέση, χρησιμοποιώντας το διαδίκτυο.
- Ασφάλεια - οι web εφαρμογές εκτίθενται σε περισσότερους κινδύνους ασφαλείας από τις εφαρμογές επιφάνειας εργασίας. Στις αυτόνομες desktop εφαρμογές, μπορεί να υπάρξει πλήρης έλεγχος και έτσι να προφυλάσσονται τα τρωτά σημεία τους. Αυτό δεν ισχύει για τις εφαρμογές web δεδομένου ότι υπάρχει πρόσβαση σε μεγάλο αριθμό χρηστών στην κοινότητα του διαδικτύου, διευρύνοντας έτσι την απειλή.

- Συνδεσιμότητα – η ανάπτυξη εφαρμογών web βασίζεται σε μεγάλο βαθμό από τη σύνδεση στο διαδίκτυο και την ταχύτητα. Η απουσία του διαδικτύου ή η κακή σύνδεση του μπορεί να προκαλέσει ζητήματα επιδόσεων με web εφαρμογές. Οι εφαρμογές desktop είναι αυτόνομες εκ φύσεως και ως εκ τούτου δεν αντιμετωπίζουν οποιαδήποτε εμπόδια που προκύπτουν από τη σύνδεση στο διαδίκτυο. Η δυνατότητα σύνδεσης επηρεάζει επίσης σημαντικά την ταχύτητα με την οποία λειτουργούν οι desktop και οι web εφαρμογές.
- Συντελεστής Κόστους – η ανάπτυξη εφαρμογών web και η συντήρησή τους περιλαμβάνουν υψηλότερο κόστος και κυρίως επαναλαμβανόμενο χαρακτήρα. Οι εφαρμογές desktop αγοράζονται μία φορά και δεν υπάρχουν συνεχώς χρεώσεις.

Βασικά πλεονεκτήματα ενός web app σε σχέση με ένα desktop app:

- μόνο ένα αντίγραφο του προγράμματος χρειάζεται ενημέρωση
- πιο εύκολη η συντήρηση, η υποστήριξη και οι ενημερώσεις
- λιγότερες πιθανότητες να υπάρχουν περιορισμοί που να σχετίζονται με τον client υπολογιστή
- και πολλά άλλα.

Βασικά μειονεκτήματα ενός web app σε σχέση με ένα desktop app:

- Εάν η σύνδεση στο δίκτυο πέσει, το ίδιο και η εφαρμογή
- Χρειάζεται η εκμάθηση πολλών γλωσσών για την ανάπτυξη μιας εφαρμογής με επιτυχία (server side scripting, SQL, XHTML/CSS κτλ.)
- Λιγότερη ασφάλεια

Όλα αυτά είναι μερικά μόνο από τα πολλά υπέρ και τα κατά που μπορούν να αναφερθούν σε αυτή τη συζήτηση.

1.1.3 Πλεονεκτήματα Frameworks για ανάπτυξη Web εφαρμογών

Στη συνέχεια παρατίθενται τα βασικά πλεονεκτήματα των frameworks για web εφαρμογές.

- Οι λειτουργίες του προγράμματος και η λογική διαχωρίζονται από τα αρχεία HTML, CSS και τα αρχεία σχεδιασμού. Αυτό βοηθά τους σχεδιαστές (χωρίς καμία εμπειρία προγραμματισμού) να είναι σε θέση να επεξεργαστούν το

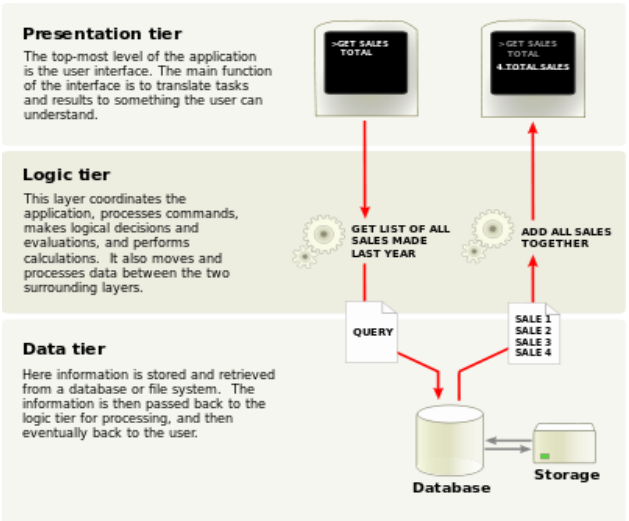
περιβάλλον και να κάνουν αλλαγές στο σχεδιασμό, χωρίς βοήθεια από έναν προγραμματιστή.

- Το χτίσιμο (builds) βασίζεται στην έννοια της μονάδας (module) και τις βιβλιοθήκες, επιτρέποντας στους προγραμματιστές να μοιράζονται εύκολα βιβλιοθήκες και να υλοποιούν πολύπλοκες λειτουργίες με έναν γρήγορο και αποτελεσματικό τρόπο.
- Η δομή βοηθά στην παραγωγή των βέλτιστων πρακτικών κωδικοποίησης με συνεπή λογική και πρότυπα κωδικοποίησης και παρέχει σε άλλους προγραμματιστές τη δυνατότητα να εξοικειωθούν με τον κώδικα σε σύντομο χρονικό διάστημα.

1.1.4 Τυπικές αρχιτεκτονικές (3-tier)

Μια εφαρμογή τριών επιπέδων αποτελεί ένα πρόγραμμα που είναι οργανωμένο σε τρία βασικά μέρη, καθένα από τα οποία διανέμεται σε μια διαφορετική θέση ή θέσεις σε ένα δίκτυο. Τα τρία μέρη είναι:

- Το περιβάλλον εργασίας ή παρουσίαση *(presentation)*
- Η επιχειρηματική λογική *(logic)*
- Η βάση δεδομένων και ο προγραμματισμός που σχετίζεται με τη διαχείρισή της (*data*)



Εικόνα 1: Επισκόπηση της αρχιτεκτονικής των τριών επιπέδων (3-tier).

Σε μια τυπική εφαρμογή 3 επιπέδων, ο σταθμός εργασίας του χρήστη της εφαρμογής περιλαμβάνει τον προγραμματισμό που παρέχει το γραφικό περιβάλλον εργασίας χρήστη (GUI) και τις φόρμες για συγκεκριμένες εφαρμογές ή διαδραστικά παράθυρα.

Η επιχειρηματική λογική βρίσκεται υλοποιημένη σε ένα διακομιστή. Η επιχειρηματική λογική υλοποιείται και εκτελείται στον διακομιστή που εξυπηρετεί τα αιτήματα των πελατών από τους σταθμούς εργασίας. Με τη σειρά του ο διακομιστής της εφαρμογής (application server), καθορίζει τι δεδομένα χρειάζεται (και πού βρίσκονται) και λειτουργεί ως πελάτης σε σχέση με ένα τρίτο επίπεδο προγραμματισμού βάσεων δεδομένων που μπορεί να βρίσκεται σε έναν κεντρικό υπολογιστή (mainframe).

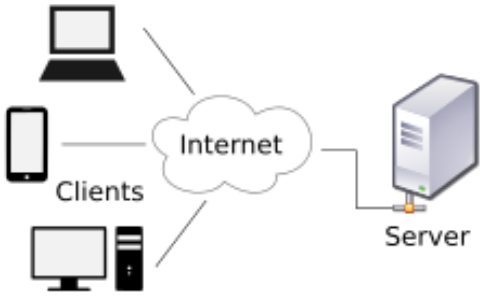
Το τρίτο επίπεδο περιλαμβάνει τη βάση δεδομένων και ένα πρόγραμμα για τη διαχείρισή της (ανάγνωση, εγγραφή κτλ). Ενώ η οργάνωση της εφαρμογής μπορεί να είναι πιο περίπλοκη από αυτό, το 3-tier μοντέλο είναι ένας βολικός τρόπος για να χωρίζονται τα τμήματα σε ένα πρόγραμμα μεγάλης κλίμακας.

Μια εφαρμογή 3-tier χρησιμοποιεί το μοντέλο πελάτη / εξυπηρετητή (client / server). Με τρεις βαθμίδες ή μέρη, κάθε μέρος μπορεί να αναπτυχθεί ταυτόχρονα από διαφορετικές ομάδες προγραμματιστών κωδικοποιώντας σε διαφορετικές γλώσσες από τους άλλους προγραμματιστές βαθμίδας. Επειδή ο προγραμματισμός για μια βαθμίδα μπορεί να αλλάξει ή να μεταφερθεί χωρίς να επηρεάζει τις άλλες βαθμίδες, το μοντέλο 3-tier καθιστά πιο εύκολο για μια επιχείρηση ή τον κατασκευαστή λογισμικού να εξελίσσει συνεχώς μια εφαρμογή καθώς προκύπτουν νέες ανάγκες και ευκαιρίες.

1.1.5 Το μοντέλο Client-Server

Το μοντέλο πελάτη-εξυπηρετητή (client–server) στην επιστήμη της πληροφορικής είναι μια κατανεμημένη δομή που διαχωρίζει τα καθήκοντα ή το φόρτο εργασίας μεταξύ του παρόχου ενός πόρου ή μιας υπηρεσίας, που ονομάζεται διακομιστής ή εξυπηρετητής (server), και των αιτούντων των υπηρεσιών, που ονομάζεται πελάτες (clients). Συχνά οι πελάτες και οι διακομιστές έχουν διαφορετικό hardware και επικοινωνούν μέσω ενός δικτύου ηλεκτρονικών υπολογιστών, αλλά είναι πιθανό να βρίσκονται και στο ίδιο σύστημα. Ένας server host τρέχει ένα ή περισσότερα προγράμματα εξυπηρετητή. ο οποίος μοιράζεται τους πόρους τους με τους πελάτες. Ένας πελάτης δεν μοιράζεται κανέναν από τους πόρους του, αλλά ζητά το

περιεχόμενο του διακομιστή ή τη λειτουργία υπηρεσιών. Ως εκ τούτου, οι πελάτες ξεκινούν συνεδρίες επικοινωνίας με διακομιστές που περιμένουν εισερχόμενες αιτήσεις.



Εικόνα 2: Διάγραμμα ενός υπολογιστικού δικτύου, όπου οι πελάτες (clients) επικοινωνούν με τον εξυπηρετητή (server) μέσω του Διαδικτύου.

1.1.6 Ο ρόλος του πελάτη και του εξυπηρετητή

Η φύση του μοντέλου client - server περιγράφει τη σχέση των προγραμμάτων που συνεργάζονται σε μια εφαρμογή. Το στοιχείο του διακομιστή παρέχει μια λειτουργία ή μια υπηρεσία σε έναν ή πολλούς πελάτες, οι οποίοι κάνουν αιτήματα για τις υπηρεσίες αυτές.

Οι διακομιστές είναι ταξινομημένοι ανάλογα με τις υπηρεσίες που παρέχουν. Για παράδειγμα, ένας διακομιστής αρχείων εξυπηρετεί παρέχοντας αρχεία του υπολογιστή. Ένας κοινόχρηστος πόρος μπορεί να είναι οποιοδήποτε λογισμικό και ηλεκτρονικά εξαρτήματα του υπολογιστή server: από προγράμματα και δεδομένα έως επεξεργαστές και συσκευές αποθήκευσης. Η κατανομή των πόρων της επιχείρησης στον server συνιστά μια υπηρεσία.

Αν ένας υπολογιστής είναι πελάτης, διακομιστής, ή και τα δύο, καθορίζεται από τη φύση της εφαρμογής που απαιτεί τις λειτουργίες της υπηρεσίας. Για παράδειγμα, ένας υπολογιστής μπορεί να τρέξει τα λογισμικά web server και file server ταυτόχρονα για να εξυπηρετήσει διαφορετικά δεδομένα για τους πελάτες που κάνουν διαφορετικά είδη αιτήσεων. Το λογισμικό πελάτη μπορεί επίσης να επικοινωνεί με το λογισμικό του server μέσα στο ίδιο υπολογιστή. Η επικοινωνία μεταξύ των servers, όπως για το συγχρονισμό δεδομένων, μερικές φορές ονομάζεται inter-server ή server-to-server επικοινωνία.

1.1.7 Τεχνολογίες Client-Side

Στο σημείο αυτό, είναι χρήσιμο να δούμε τον τρόπο λειτουργίας των τεχνολογιών πελάτη (Client-Side), και στη συνέχεια παρατίθενται κάποιες γνωστές εφαρμογές-πελάτη.

Ο κώδικας script από την πλευρά του πελάτη (Client Side Scripting / Coding) είναι το είδος του κώδικα που εκτελείται ή ερμηνεύεται από τους browsers.

Το Client Side Scripting είναι γενικά ορατό από κάθε επισκέπτη σε μια τοποθεσία (από το μενού προβολής στον browser, με κλικ στο "προβολή πηγής" μπορεί κάποιος να δει τον πηγαίο κώδικα).

Παρακάτω είναι μερικές συνηθισμένες Client Side Scripting τεχνολογίες:

- HTML (HyperText Markup Language)
- CSS (Cascading Style Sheets)
- JavaScript
- Ajax (Asynchronous JavaScript and XML)
- jQuery (JavaScript Framework Library - που χρησιμοποιούνται συνήθως στην ανάπτυξη της Ajax)
- MooTools (JavaScript Framework Library - που χρησιμοποιούνται συνήθως στην ανάπτυξη της Ajax).

1.1.8 Τεχνολογίες Server-Side

Ο κώδικας script από την πλευρά του εξυπηρετητή (Server Side Scripting / Coding) είναι το είδος του κώδικα που εκτελείται ή ερμηνεύεται από τον web server.

Το Server Side Scripting δεν είναι ορατό ή προσβάσιμο από οποιονδήποτε επισκέπτη ή το ευρύτερο κοινό.

Παρακάτω παρατίθενται οι συνηθισμένες Server Side Scripting τεχνολογίες:

- PHP (πολύ κοινή γλώσσα για Server Side Scripting, ανοικτού κώδικα βασισμένη σε Linux / Unix, η διανομή της είναι δωρεάν και συνήθως συνδυάζεται με βάσεις δεδομένων MySQL)
- Zend Framework (αντικειμενοστραφές framework της PHP για διαδικτυακές εφαρμογές)
- ASP (Script Γλώσσα Microsoft Web Server (IIS))

- ASP.NET (Framework για διαδικτυακές εφαρμογές της Microsoft και διάδοχος της ASP)
- ColdFusion (Framework για διαδικτυακές εφαρμογές της Adobe)
- Ruby on Rails (Framework για διαδικτυακές εφαρμογές με προγραμματισμό, δωρεάν διανομής)
- Perl (υψηλού επιπέδου γλώσσα προγραμματισμού γενικού σκοπού, που χρησιμοποιείται και ως γλώσσα script server-side, διανέμεται δωρεάν, αλλά έχασε τη δημοτικότητά της λόγω της PHP)
- Python (υψηλού επιπέδου γλώσσα προγραμματισμού γενικού σκοπού, που χρησιμοποιείται και ως γλώσσα script server-side, διανέμεται δωρεάν).

1.2 Σχεδίαση / Κύκλοι ανάπτυξης Web εφαρμογής

1.2.1 Μοντέλα Κύκλου Ζωής των Web εφαρμογών

Ο κύκλος ζωής των web εφαρμογών είναι η διαδικασία της ανάπτυξης μιας web εφαρμογής και η εμπλοκή των πολλαπλών ομάδων που ασχολούνται με την αναπτυξιακή διαδικασία. Κάθε οργανισμός μπορεί να ορίζει το δικό του μοναδικό στυλ λειτουργίας.

Ορισμένες εταιρείες ακολουθούν ένα συγκεκριμένο πρότυπο μοντέλο, όπως το SDLC (System Development Life Cycle) ή το ευέλικτο Agile Software Development Model.

- SDLC είναι η παραδοσιακή διαδικασία της ανάπτυξης λογισμικού ή web εφαρμογών που περιλαμβάνει την έρευνα για τον εντοπισμό και καθορισμό των απαιτήσεων της εφαρμογής, την ανάλυση των πληροφοριών, τον αρχιτεκτονικό σχεδιασμό, τις προδιαγραφές του προσχεδίου, την ομαδική συμμετοχή, τον προγραμματισμό, τον έλεγχο και την διόρθωση σφαλμάτων, τον έλεγχο του συστήματος, την εφαρμογή και τη συντήρηση.
- Ευέλικτη (Agile) Ανάπτυξη Λογισμικού / Διαδικτυακών Εφαρμογών περιλαμβάνει την έρευνα, την ανάλυση, τη διαχείριση του έργου, το σχεδιασμό, τον προγραμματισμό, την υλοποίηση, τη συχνή δοκιμή, την προσαρμογή και συντήρηση.

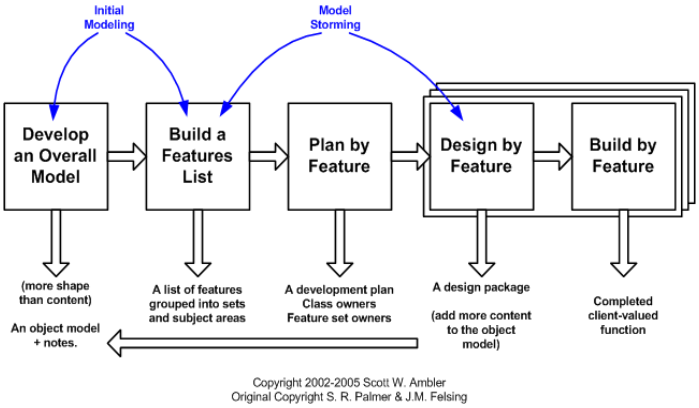
Σε μια διαδικτυακή επιχείρηση, τα τρία σημαντικότερα θέματα (χρόνος διάθεσης στην αγορά, ανάπτυξη της εταιρίας και ανάλυση απαιτήσεων), συμπίπτουν με τις

αρχές των ευέλικτων (agile) πρακτικών. Κάποια ευέλικτα μοντέλα κύκλου ζωής είναι:

- Ακραίος Προγραμματισμός (Extreme programming). Πρόκειται για την πιο δημοφιλή από τις Ευέλικτες Μεθόδους και δημιουργήθηκε στα τέλη της δεκαετίας του '90 από τον Kent Beck. Είναι μία πειθαρχημένη προσέγγιση που υποστηρίζει την ταχύτερη ολοκλήρωση του έργου σε μικρούς επαναληπτικούς κύκλους, επιτρέποντας στους υπεύθυνους ανάπτυξης να ανταποκρίνονται στις μεταβαλλόμενες απαιτήσεις του πελάτη καθ' όλη τη διάρκεια του κύκλου παραγωγής (Beck, 2000).
- Το πλαίσιο Scrum**.** Το Scrum είναι ένα πλαίσιο διαδικασιών που έχει χρησιμοποιηθεί για την διαχείριση της ανάπτυξης πολύπλοκων προϊόντων από τις αρχές της δεκαετίας του 1990. Το Scrum θεμελιώνεται στη θεωρία του εμπειρικού ελέγχου διαδικασιών, αλλιώς εμπειρισμό. Ο εμπειρισμός υποστηρίζει ότι η γνώση προέρχεται από την εμπειρία και από τη λήψη αποφάσεων βασισμένων σε ό, τι γνωρίζουμε.
- Ανάπτυξη Timebox. Ένα πλαίσιο χρόνου (timebox) αποτελεί μια προκαθορισμένη χρονική περίοδο κατά την οποία πρέπει ένα έργο να έλθει σε πέρας.
- Ανάπτυξη με βάση τα χαρακτηριστικά (Feature-driven development – FDD). Αποτελεί μια διαδικασία ανάπτυξης λογισμικού βασισμένη στον πελάτη και την αρχιτεκτονική. Όπως φαίνεται και από το όνομα, τα χαρακτηριστικά (features) αποτελούν ένα κεντρικό θέμα στο FDD (De Luca et.al., 1999). Σαν χαρακτηριστικό αναφέρεται μια μικρή λειτουργία του πελάτη, που εκφράζεται στη μορφή <action><result><object>.

Όπως φαίνεται στην Εικόνα 3, υπάρχουν πέντε κύριες δραστηριότητες που λαμβάνουν χώρα στο FDD με επαναλαμβανόμενο τρόπο:

- Ανάπτυξη ενός γενικού μοντέλου
- Σχεδιασμός και ομαδοποίηση λίστας γνωρισμάτων
- Σχεδιασμός και ανάπτυξη βάσει των παραπάνω γνωρισμάτων
    1. Εισαγωγή επιπλέον περιεχομένου και χαρακτηριστικών
    2. Ολοκλήρωση μοντέλου



Εικόνα 3: Κύκλος ζωής έργου στο μοντέλο FDD.

1.2.2 Η διαδικασία ανάπτυξης των web εφαρμογών

Η διαδικασία ανάπτυξης των web εφαρμογών οργανώνει μια πρακτική διαδικασία και προσέγγιση στην ανάπτυξη εφαρμογών.

Η παρακάτω λίστα των διαδικασιών και των προτεινόμενων εγγράφων παρέχει ένα καλό περίγραμμα για τον Κύκλο Ζωής μιας Web Εφαρμογής:

- Έγγραφο χαρτογράφησης έργου (Roadmap): Καθορίζει τον σκοπό, στόχους και κατεύθυνση της Web εφαρμογής
- Έγγραφο ομαδική συνεργασίας και διαχείρισης έργου
- Επιλογή τεχνολογίας, τεχνικές προδιαγραφές, οπτικό διάγραμμα αρχιτεκτονικής και δομής της web εφαρμογής, μεθοδολογία ανάπτυξης, έλεγχος εκδόσεων, αντίγραφα ασφαλείας, αναβαθμίσεις, έγγραφο επεκτάσεων και σχεδιασμού ανάπτυξης, επιλογή υλικού και λογισμικού server
- Ανάλυση και επιλογή εμπορικών ζητημάτων (οικονομικά θέματα, πιστοποιητικό SSL, πάροχος και διαχειριστής server, λογισμικό για analytics στατιστικά των επισκεπτών του site, συστήματα πληρωμών τρίτων κτλ.)
- Οπτικός οδηγός εφαρμογής, σχεδιασμός Layout, σχεδιασμός διεπαφής
- Σχεδιασμός βάσης δεδομένων και ανάπτυξη web εφαρμογής
- Έλεγχος (testing): Διασφάλιση ποιότητας, συμβατότητα πολλαπλών browser, ασφάλεια, απόδοση – τεστ φόρτου (load) και stress test, χρηστικότητα
- Συντήρηση.

1.2.3.Έλεγχος (testing) των web εφαρμογών

Η δοκιμή είναι σημαντικό κομμάτι της διαδικασίας ανάπτυξης των web εφαρμογών. Κατά περίσταση, οι δοκιμές μπορεί να καταναλώσουν περισσότερο ανθρώπινο δυναμικό και χρόνο από την ίδια την ανάπτυξη.

Οι web εφαρμογές υφίστανται τους ίδιους ελέγχους – δοκιμές με τις παραδοσιακές desktop εφαρμογές. Αλλά επειδή οι πελάτες web εφαρμογών ποικίλουν αρκετά, μπορούν να εκτελεστούν και κάποιοι πρόσθετοι έλεγχοι, όπως:

- Ασφάλειας
- Απόδοσης
- Επικύρωση HTML/CSS
- Προσβασιμότητας
- Χρηστικότητας
- Συμβατότητα πολλαπλών browser

Συμπέρασμα

Το συγκεκριμένο κεφάλαιο μας βοήθησε να αποσαφηνίσουμε τα σημαντικότερα θέματα αναφορικά με τις web εφαρμογές καθώς αυτό κρίνεται πολύ βασικό πριν προχωρήσουμε στον κύριο κορμό της παρούσας εργασίας.

Κεφάλαιο 2
CMS

Στο κεφάλαιο 2 πρόκειται να μιλήσουμε για τις τεχνολογίες server side που υπάρχουν. Στη συνέχεια παραθέτουμε τις απόψεις ορισμένων ειδικών σχετικά με την επιλογή γλώσσας προγραμματισμού που χρησιμοποιείται πιο συχνά, ενώ στη συνέχεια αναφέρουμε κάποια θέματα για τα εργαλέια ανάπτυξης διαδικτυακών εφαρμογών και για τα συστήματα διαχείρισης περιεχομένου CMS.

2.1 Τι τεχνολογίες server side υπάρχουν

Υπάρχουν πολλές σύγχρονες τεχνολογίες και γλώσσες για την ανάπτυξη διαδικτυακών εφαρμογών. Οι σημαντικότεροι εκπρόσωποί τους είναι οι εξής: Ajax, ASP, ASP.NET, ActionScript, CSS, HTML, Java, PHP , Python και Ruby.

Ajax: Με την Ajax οι web εφαρμογές είναι δυνατόν να στέλνουν και να ανακτούν δεδομένα από ένα διακομιστή, χωρίς να πραγματοποιείται κάποια αλλαγή στο URL της τρέχουσας ιστοσελίδας. Η Ajax δίνει τη δυνατότητα στις σελίδες και στις web εφαρμογές να να επεκτείνουν και να αλλάζουν το περιεχόμενων μιας σελίδας δυναμικά, χωρίς να χρειάζεται να φορτώσουν μια επιπλέον σελίδα.

ASP, ASP.NET: θα αναφερθούμε αναλυτικά στο κεφάλαιο 3.

ActionScript: πρόκειται για μια γλώσσα προγραμματισμού, η οποία χρησιμοποιείται με σκοπό την ανάπτυξη τόσο ιστοσελίδων όσο και λογισμικού για την πλατφόρμα Adobe Flash Player.

Java Server Pages: επιτρέπει στον προγραμματιστή να εμφωλεύει κώδικα java σε κώδικα HTML με την χρήση μιας σύνταξης ανάλογης με αυτών της php και asp. Επίσης, χρησιμοποιεί ένα σύνολο XML tags για να ορίσει συγκεκριμένες λειτουργίες.

PHP: είναι μια γλώσσα προγραμματισμού open source server-side και δημιουργεί δυναμικές ιστοσελίδες. Ο κώδικας της συγκεκριμένης γλώσσας ερμηνεύεται από τον web server καλώντας έναν επεξεργαστή PHP που δημιουργεί την ιστοσελίδα που έχει δημιουργηθεί.

Python: είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου και έχει ως βασικό στόχο την αναγνωσιμότητα του κώδικά της. Η ευκολία της χρήσης της καθώς και το συντακτικό της επιτρέπει στους προγραμματιστές να δημιουργήσουν έννοιες σε λιγότερες γραμμές κώδικα

Ruby: πρόκειται για μια δυναμική γλώσσα προγραμματισμού η οποία εστιάζει στην απλότητα και την παραγωγικότητα ταυτόχρονα καθώς προσφέρει ευκολία τόσο στη σύνταξη όσο και στην ανάγνωση κώδικα.

2.2 Εργαλεία για ανάπτυξη διαδικτυακών εφαρμογών

Στην περίπτωση της ASP.NET, ο προγραμματιστής μπορεί να χρησιμοποιήσει το Microsoft Visual Studio για να γράψει κώδικα. Αλλά, όπως συμβαίνει με τις περισσότερες άλλες γλώσσες προγραμματισμού, μπορεί επίσης να χρησιμοποιηθεί ένα πρόγραμμα επεξεργασίας κειμένου. Το Notepad++ είναι ένα παράδειγμα.

Για την συγγραφή ανοικτού κώδικα υπάρχουν πολλά διαθέσιμα εργαλεία για τη σύνταξη κώδικα. Αυτά περιλαμβάνουν το Adobe Dreamweaver CS4, το plugin CFEclipse για το λογισμικό Eclipse και το Adobe CF Builder. Μπορεί επίσης να χρησιμοποιηθεί και πάλι οποιοδήποτε πρόγραμμα επεξεργασίας κειμένου, όπως το Notepad ++ ή το TextEdit.

Για την Java (γλώσσα προγραμματισμού), υπάρχουν πολλά εργαλεία συγγραφής και μεταγλώττισης κώδικα. Τα πιο δημοφιλή είναι το Apache Tomcat, GlassFish, JDeveloper και Netbeans, αλλά υπάρχουν και πολλά άλλα.

Για την PHP, το Zend Development Environment παρέχει πολλά εργαλεία εντοπισμού σφαλμάτων και παρέχει πλούσια χαρακτηριστικά που μπορούν να κάνουν τη ζωή ενός προγραμματιστή PHP ευκολότερη.

2.3 Συστήματα Διαχείρισης Περιεχομένου - Content Management Systems (CMS)

Τα Συστήματα Διαχείρισης Περιεχομένου - Content Management Systems (CMS) αναφέρονται στις εφαρμογές οι οποίες δίνουν τη δυνατότητα στον πελάτη να διαχειρίζεται το δικτυακό του περιεχόμενο, όπως κείμενα, εικόνες, πίνακες κ.λπ., με εύκολο τρόπο, συνήθως παρόμοιο με αυτόν της χρήσης ενός κειμενογράφου. Οι εφαρμογές διαχείρισης περιεχομένου επιτρέπουν την αλλαγή του περιεχόμενου χωρίς να είναι απαραίτητες ειδικές γνώσεις σχετικές με τη δημιουργία ιστοσελίδων ή γραφικών, καθώς συνήθως τα κείμενα γράφονται μέσω κάποιων online WYSIWYG

("What You See Is What You Get") html editors, ειδικών δηλαδή κειμενογράφων, παρόμοιων με το MS Word, που επιτρέπουν τη μορφοποίηση των κειμένων όποτε υπάρχει ανάγκη.

Οι αλλαγές του site μπορούν να γίνουν από οποιονδήποτε υπολογιστή που είναι συνδεμένος στο Διαδίκτυο, χωρίς να χρειάζεται να έχει εγκατεστημένα ειδικά προγράμματα επεξεργασίας ιστοσελίδων, γραφικών κ.λπ. Μέσω ενός απλού φυλλομετρητή ιστοσελίδων (browser), ο χρήστης μπορεί να συντάξει ένα κείμενο και να ενημερώσει άμεσα το δικτυακό του τόπο.

Αυτό που αποκαλούμε πολλές φορές "δυναμικό περιεχόμενο" σε ένα website δεν είναι άλλο παρά οι πληροφορίες που παρουσιάζονται στο site και μπορούν να αλλάξουν από τους ίδιους τους διαχειριστές του μέσω κάποιας εφαρμογής, η οποία ουσιαστικά μπορεί να εισάγει (προσθέτει), διορθώνει και να διαγράφει εγγραφές σε πίνακες βάσεων δεδομένων, όπου τις περισσότερες φορές καταχωρούνται όλες αυτές οι πληροφορίες.

Αυτό σημαίνει ότι δεν χρειάζεται να δημιουργηθούν πολλές ξεχωριστές ιστοσελίδες για την παρουσίαση των πληροφοριών στο site, αλλά αρκεί ένας ενιαίος σχεδιασμός στα σημεία όπου θέλουμε να εμφανίζεται το περιεχόμενό μας, καθώς και να υπάρχει ο ειδικός σε κάποια συγκεκριμένη γλώσσα προγραμματισμού (ASP, PHP, Coldfusion, Perl, CGI κ.λπ.), ο οποίος αναλαμβάνει να εμφανίσει τις σωστές πληροφορίες στις σωστές θέσεις.

Έτσι, για το δικτυακό τόπο μιας εφημερίδας π.χ., που απαιτεί εύλογα καθημερινή ενημέρωση αλλά δεν χρησιμοποιεί κάποιο σύστημα Content Management, θα πρέπει ο υπεύθυνος για το σχεδιασμό του (designer) να δημιουργήσει μια σελίδα με τα γραφικά, την πλοήγηση και το περιβάλλον διεπαφής (interface) του website, ο υπεύθυνος ύλης να τοποθετήσει το περιεχόμενο στα σημεία της ιστοσελίδας που θέλει, και να ενημερωθούν οι σύνδεσμοι των υπόλοιπων σελίδων ώστε να συνδέονται με την καινούργια. Αφού την αποθηκεύσει, πρέπει να την ανεβάσει στο website μαζί με τις υπόλοιπες ιστοσελίδες που άλλαξαν.

Αντιθέτως, αν ο δικτυακός τόπος λειτουργεί με χρήση κάποιου συστήματος CM, το μόνο που έχει να κάνει ο διαχειριστής του είναι να ανοίξει τη σχετική φόρμα εισαγωγής νέου άρθρου στη διαχειριστική εφαρμογή του website και να γράψει ή να επικολλήσει (copy-paste) τα στοιχεία που επιθυμεί. Αυτόματα, μετά την καταχώριση γίνονται από το ίδιο το σύστημα διαχείρισης περιεχομένου όλες οι απαραίτητες

ενέργειες, ώστε το άρθρο να είναι άμεσα διαθέσιμο στους επισκέπτες και όλοι οι σύνδεσμοι προς αυτό ενημερωμένοι.

Με την αυξητική τάση χρήσης των CMS στην Ελλάδα και το εξωτερικό, γίνεται εμφανές ότι το μέλλον του Διαδικτύου σε ό,τι αφορά περιεχόμενο και πληροφορίες που πρέπει να ανανεώνονται τακτικά, ανήκει στα προγράμματα διαχείρισης περιεχομένου, αφού προσφέρουν πολλά πλεονεκτήματα, ταχύτητα και ευκολίες στη χρήση τους.

Τα Συστήματα Διαχείρισης Περιεχομένου μπορούν να χρησιμοποιηθούν και να αντικαταστήσουν ένα συμβόλαιο συντήρησης επάξια. Τα CMS μπορούν να χρησιμοποιηθούν για:

- Ειδήσεις (εφημερίδες, περιοδικά, πρακτορεία ειδήσεων κ.λπ.)
- Παρουσιάσεις εταιριών και προσωπικού
- Καταλόγους προϊόντων
- Παρουσιάσεις προϊόντων
- Online υποστήριξη
- Αγγελίες και ανακοινώσεις
- Παρουσιάσεις και προβολή γεωγραφικών περιοχών
- Διαφημίσεις
- Δελτία Τύπου
- Όρους και συμβόλαια
- Χάρτες, κατευθύνσεις, οδηγίες

Ένα ολοκληρωμένο CMS πρέπει να μπορεί να διαχειρίζεται όλες τις δυναμικές πληροφορίες του site και να προσφέρει υπηρεσίες που εξυπηρετούν πλήρως τις ανάγκες των διαχειριστών του.

Επιγραμματικά, μερικά από τα πλεονεκτήματα και τα χαρακτηριστικά ενός ολοκληρωμένου CMS είναι:

- Γρήγορη ενημέρωση, διαχείριση και αρχειοθέτηση του περιεχομένου του δικτυακού τόπου
- Ενημέρωση του περιεχομένου από οπουδήποτε
- Ταυτόχρονη ενημέρωση από πολλούς χρήστες και διαφορετικούς υπολογιστές
- Να μην απαιτούνται ειδικές τεχνικές γνώσεις από τους διαχειριστές του

- Εύκολη χρήση και άμεση γνώση του τελικού αποτελέσματος, όπως γίνεται με τους γνωστούς κειμενογράφους
- Δυνατότητα αναζήτησης του περιεχομένου που καταχωρείται και αυτόματη δημιουργία αρχείου
- Ασφάλεια και προστασία του σχεδιασμού του site από λανθασμένες ενέργειες, που θα μπορούσαν να δημιουργήσουν προβλήματα στην εμφάνισή του
- Διαχωρισμός του περιεχομένου από το σχεδιασμό και την πλοήγηση (navigation) του δικτυακού τόπου
- Αλλαγή σχεδιασμού ή τρόπου πλοήγησης χωρίς να είναι απαραίτητη η ενημέρωση όλων των σελίδων από τον ίδιο το χρήστη
- Αυτόματη δημιουργία των συνδέσμων μεταξύ των σελίδων και αποφυγή προβλημάτων ανύπαρκτων σελίδων (404 error pages)
- Μικρότερος φόρτος στον εξυπηρετητή (server) και χρήση λιγότερου χώρου, αφού δεν υπάρχουν πολλές επαναλαμβανόμενες στατικές σελίδες, από τη στιγμή που η ανάπτυξη των σελίδων γίνεται δυναμικά
- Όλο το περιεχόμενο καταχωρείται στην/στις βάσεις δεδομένων, τις οποίες μπορούμε πιο εύκολα και γρήγορα να τις προστατεύσουμε τηρώντας αντίγραφα ασφαλείας.

Drupal

Το Drupal είναι σίγουρα η πιο σύνθετη πλατφόρμα από τις τρεις και απευθύνεται σε χρήστες με περισσότερες τεχνικές γνώσεις. Ωστόσο, παρέχει τη δυνατότητα ανάπτυξης πολυπλοκότερων ιστοσελίδων συγκριτικά με τις άλλες δύο πλατφόρμες. Εάν δεν είστε διατεθειμένοι να ασχοληθείτε αρκετές ώρες με την εκμάθηση αυτού του CMS, ίσως να μην είναι η καλύτερη επιλογή για σας.

Μερικά από τα σημαντικότερα πλεονεκτήματα του Drupal είναι τα εξής:

Τεχνικά προηγμένο: Το Drupal είναι το πιο “technically advanced” από τα τρία CMS. Ο τρόπος που έχει αναπτυχθεί του επιτρέπει να χρησιμοποιεί πολύ λιγότερους πόρους στον server από το WordPress και το Joomla.

Βελτιωμένη απόδοση: Ιστοσελίδες βασισμένες στο Drupal, φορτώνουν γρηγορότερα και παρουσιάζουν χαμηλότερους χρόνους απόκρισης σε σύγκριση με τα άλλα δύο CMS.

Παραμετροποιησιμότητα: Με μεγάλο αριθμό διαθέσιμων plug-ins, αλλά και τη δυνατότητα παρέμβασης στον πηγαίο κώδικα, το Drupal είναι το πιό ευέλικτο από τα τρία CMS σε επίπεδο παραμετροποιησιμότητας.

Ευέλικτο: Χρειάζεστε ένα απλό one-page blog; Μπορείτε να χρησιμοποιήσετε το Drupal.

Το Drupal έιναι ένα ισχυρό εργαλείο διαχείρισης περιεχομένου. Όπως όλα τα ισχυρά εργαλεία, έτσι και το Drupal απαιτεί ιδιαίτερες γνώσεις προγραμματισμού και αρκετή εμπειρία, που οι μέσοι χρήστες δεν διαθέτουν. Αυτός ο παράγοντας, καθιστά την εύρεση υποστήριξης δυσκολότερη από ότι στην περίπτωση των δύο άλλων CMS. Εάν ανήκετε στην κατηγορία του μέσου χρήστη, καλό θα ήταν να αποφύγετε αυτή την επιλογή.

Joomla

Το Joomla είναι ένα αρκετά δυνατό εργαλείο διαχείρισης περιεχομένου που μπορεί να προσφέρει πολλά από τα χαρακτηριστικά του Drupal, χωρίς όμως να απαιτεί τόσο μεγάλη τεχνική εμπειρία και γνώση. Όπως και στις άλλες δύο περιπτώσεις, έτσι και στην περίπτωση του Joomla, υπάρχεί μεγάλο πλήθος από plug-ins και themes για να επιλέξετε και να παραμετροποιήσετε την ιστοσελίδα σας όπως εσείς επιθυμείτε.

Οι βασικοί λόγοι που οι χρήστες επιλέγουν το Joomla είναι οι εξής:

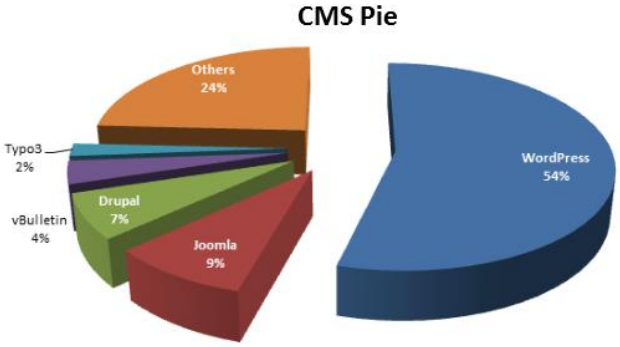
Commerce Sites: Είναι το ιδανικότερο από τα τρία CMS για τη δημιουργία ηλεκτρονικών καταστημάτων. Παρόλο που αυτή η δυνατότητα παρέχεται και από τις άλλες δύο πλατφόρμες, με το Joomla μπορείτε να φτιάξετε ένα e-shop ταχύτερα και πιο εύκολα λόγω της καλύτερης υποστήριξης που προσφέρει για τέτοιου είδους ιστοσελίδες.

Δεν απαιτεί πολλές τεχνικές γνώσεις: Το Joomla, κατά γενική ομολογία αποτελεί τη μέση λύση ανάμεσα στην ευκολία διαχείρισης που προσφέρει το WordPress και στη "δύναμη" που χαρακτηρίζει ένα Drupal-powered website. Ο μέσος χρήστης έχει τη δυνατότητα να διαχειριστεί ένα Joomla-powered site, χωρίς να έχει ιδιαίτερες τεχνικές γνώσεις.

Υποστήριξη: Το Joomla διαθέτει μια αρκετά μεγάλη κοινότητα υποστήριξης που παρέχει λύσεις και απαντήσεις σε απλά αλλά και πολύπλοκα θέματα και ερωτήματα που μπορεί να προκύψουν για το μέσο χρήστη. Μικρότερη μεν από αυτή του WordPress, αρκετά μεγαλύτερη όμως από αυτή του Drupal.

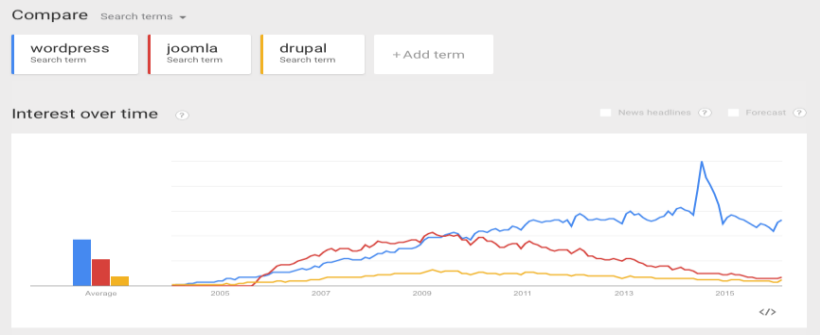
Όπως το Drupal, έτσι και το Joomla απαιτεί αρκετές τεχνικές γνώσεις. Για να διαχειριστείτε σωστά την ιστοσελίδα σας, θα πρέπει να κατανοήσετε τη φιλοσοφία λειτουργίας του, πράγμα που απαιτεί αρκετές ώρες μελέτης και ενασχόλησης με την συγκεκριμένη πλατφόρμα.

Στην εικόνα που ακολουθεί μπορείτε να δείτε τα στατιστικά χρήσης μερικών δημοφιλών CMS.



Εικόνα 4: CMS Usage Analytics

Στο παρακάτω διάγραμμα μπορούμε να παρατηρήσουμε πώς κυμαίνεται το ενδιαφέρον από την χρήση των Joomla, Wordpress και drupal από το 2005 έως το 2015.



Εικόνα 5: Χρήση CMS απο το 2005-2015

Συμπέρασμα

Ο σημαντικότερος ρόλος των Συστημάτων Διαχείρισης Περιεχομένου (CMS), είναι να σχεδιάζουν ιστοσελίδες με ένα κατανοητό τρόπο, να ανανεώνουν και να ενημερώνουν μιας ιστοσελίδας ώστε να γίνεται άμεσα και χωρίς επιπλέον κόστος,το ότι δεν απαιτούν εξειδικευμένες γνώσεις ως προς τη διαχείριση μιας ιστοσελίδας με συνακόλουθο αποτέλεσμα να είναι μια πολύ εύκολη διαδικασία για ένα μέσο χρήστη και το ότι επεκτείνουν τις παρούσες δυνατότητες ανάλογα με τις απαιτήσεις που

υπάρχουν κάθε φορά. Το γεγονός ότι πρόκειται για ένα σύστημα διαχείρισης περιεχομένου το οποίο παρέχει στους χρήστες ένα γραφικό περιβάλλον το οποίο έχει τη δυνατότητα να εκτελεί διάφορες λειτουργίες όχι μόνο διαχείρισης αλλά και λειτουργίες ενός δυναμικού περιεχομένου που διαφορετικά θα απαιτούσαν χρόνο και μεγαλύτερες προσπάθειες από τους ίδιους προγραμματιστές, μας οδηγεί να συνειδητοποιήσουμε πως τα CMS αποτελούν πολύ σημαντικό παράγοντα των διαδικτυακών εφαρμογών. Αυτός είναι και ο βασικός λόγος για τον οποίο η δημιουργία ιστότοπων με δυναμικό περιεχόμενο έγινε πραγματοποιήσιμη για ένα μεγαλύτερο εύρος χρηστών με συνακόλουθο αποτέλεσμα την υπέρμετρη αύξηση δημιουργίας ιστοσελίδων.

Κεφάλαιο 3
ASP.NET

Στο κεφάλαιο 3 θα μιλήσουμε αναλυτικά για την ASP.NET, για την πλατφόρμα .NET και πιο συγκεκριμένα για την ιστορία της, την αρχιτεκτονική, τις συμβολικές γλώσσες, τα μοντέλα εφαρμογών, τις αρχές σχεδιασμού, τις εναλλακτικές υλοποιήσεις, την NET Core, καθώς και τα μοντέλα ASP.NET.

3.1 Η πλατφόρμα .NET

Το .NET Framework (προφέρεται dot net) είναι ένα πλαίσιο λογισμικού που αναπτύχθηκε από τη Microsoft και τρέχει κυρίως σε Microsoft Windows, Linux και macOS/OS X. Οι υποστηριζόμενες εκδόσεις Windows στην .NET είναι οι εξής: Windows 7SP1, Windows 8.1/10, Windows Server 2008 R2 SP1, Windows Server 2012 SP1, Windows Server 2012 R2 SP1, Windows Server 2016. Στο περιβάλλον της Linux υποστηρίζεται από τις ακόλουθες εκδόσεις: Ubuntu 14.04,16.04, Linux Mint 17, Debian 8.2, Fedora 23, Oracle Linux 7.1, openSUSE 13.2 και Linux 7 server. Όσο αφορά το περιβάλλον macOS η πλατφόρμα .NET υποστηρίζεται από το OS X El Capitan(version 10.11) και το macOS sierra (version 10.12).

Η πλατφόρμα .NET περιλαμβάνει μια μεγάλη βιβλιοθήκη κλάσεων, γνωστή ως Framework Class Library (FCL) και παρέχει διαλειτουργικότητα γλώσσας (μια γλώσσα μπορεί να χρησιμοποιήσει κώδικα γραμμένο σε άλλες γλώσσες) σε διάφορες γλώσσες προγραμματισμού.

Προγράμματα που είναι γραμμένα για το .NET Framework εκτελούνται σε ένα περιβάλλον λογισμικού (σε αντίθεση με το περιβάλλον υλικό) γνωστό ως Common Language Runtime (CLR), μια εφαρμογή εικονικής μηχανής που παρέχει υπηρεσίες, όπως η ασφάλεια και τη διαχείριση μνήμης. (Ως εκ τούτου, ο κώδικας που γράφεται σε .NET Framework ονομάζεται "διαχειριζόμενος κώδικας".) Το FCL και η CLR μαζί αποτελούν το .NET Framework.

Το FCL παρέχει user interface, πρόσβαση στα δεδομένα, σύνδεση με βάσεις δεδομένων, κρυπτογραφία, ανάπτυξη web εφαρμογών, αριθμητικούς αλγόριθμους, και επικοινωνίες δικτύου. Οι προγραμματιστές παράγουν λογισμικό συνδυάζοντας το δικό τους κώδικα με το .NET Framework και άλλες βιβλιοθήκες. Το .NET Framework προορίζεται για χρήση στις περισσότερες νέες εφαρμογές που έχουν

δημιουργηθεί για την πλατφόρμα των Windows. Η Microsoft παράγει επίσης ένα ολοκληρωμένο περιβάλλον ανάπτυξης σε μεγάλο βαθμό για το λογισμικό .NET που ονομάζεται Visual Studio.

Το .NET Framework ξεκίνησε ως ένα ιδιόκτητο πλαίσιο, αν και η εταιρεία εργάστηκε για την τυποποίηση της στοίβας λογισμικού σχεδόν αμέσως, ακόμα και πριν από την πρώτη κυκλοφορία του. Παρά τις προσπάθειες τυποποίησης, οι προγραμματιστές- ιδιαίτερα εκείνοι στις κοινότητες ελεύθερου λογισμικού και ανοικτού κώδικα - εξέφρασαν την ανησυχία τους με τους επιλεγμένους όρους και τις προοπτικές της κάθε εφαρμογής ελεύθερου και ανοιχτού κώδικα, ιδίως όσον αφορά τις πατέντες λογισμικού.

Το .NET Framework οδήγησε σε μια οικογένεια από πλατφόρμες .NET που στοχεύουν το mobile computing, τις ενσωματωμένες συσκευές, εναλλακτικά λειτουργικά συστήματα και τα plugins του προγράμματος περιήγησης. Μια μειωμένη έκδοση του πλαισίου, το .NET Framework Compact, είναι διαθέσιμο στις πλατφόρμες με Windows CE, συμπεριλαμβανομένων των συσκευών των Windows Mobile, όπως τα smartphones. Το .NET Micro Framework απευθύνεται σε ενσωματωμένες συσκευές περιορισμένων πόρων. Το Silverlight ήταν διαθέσιμο ως πρόσθετο πρόγραμμα περιήγησης στο Web. Το Mono είναι διαθέσιμο για πολλά λειτουργικά συστήματα και είναι προσαρμοσμένο σε δημοφιλή smartphone λειτουργικά συστήματα (Android και iOS) και μηχανές παιχνιδιών. Το .NET Core στοχεύει σε cross-platform και cloud-based workloads πέρα από την πλατφόρμα Universal Windows (UWP).

3.1.1 Ιστορία

Η Microsoft ξεκίνησε την ανάπτυξη του .NET Framework στα τέλη της δεκαετίας του 1990, αρχικά με το όνομα Next Generation Windows Services (NGWS). Μέχρι τα τέλη του 2000, κυκλοφόρησαν οι πρώτες εκδόσεις beta του .NET 1.0.

Τον Αύγουστο του 2000, η Microsoft, η Hewlett-Packard, και η Intel εργάστηκαν για την τυποποίηση της Common Language Infrastructure (CLI) και της C#. Μέχρι το Δεκέμβριο του 2001, και οι δύο είχαν επικυρώσει τα πρότυπα ECM. Η ISO ακολούθησε τον Απρίλιο του 2003. Η τρέχουσα έκδοση των προτύπων ISO είναι ISO/IEC 23271:2012 και ISO/IEC 23270:2006.

Ενώ η Microsoft και οι συνεργάτες τους είναι κάτοχοι διπλωμάτων ευρεσιτεχνίας για τις CLI και C#, η ECMA και η ISO απαιτούν όλα τα διπλώματα ευρεσιτεχνίας ουσιαστικής σημασίας για την εφαρμογή να γίνουν με «λογικούς και ισότιμους όρους». Εκτός από την εκπλήρωση αυτών των όρων, οι εταιρείες έχουν συμφωνήσει να διαθέσει τα διπλώματα ευρεσιτεχνίας ατελώς. Ωστόσο, αυτό δεν ισχύει για το τμήμα του .NET Framework που δεν καλύπτεται από τα πρότυπα ECMA / ISO, το οποίο περιελάμβανε τα Windows Forms, ADO.NET, και ASP.NET. Πατέντες που η Microsoft κατέχει σε αυτές τις περιοχές μπορεί να αποθαρρύνουν τις μη-Microsoft υλοποιήσεις του πλήρους πλαισίου.

Στις 3 Οκτωβρίου 2007, η Microsoft ανακοίνωσε ότι ο πηγαίος κώδικας για τις .NET Framework 3.5 βιβλιοθήκες ήταν να γίνουν διαθέσιμες στο πλαίσιο του Microsoft Reference Source License (Ms-RSL). Η αποθήκη του πηγαίου κώδικα έγινε διαθέσιμη στο διαδίκτυο στις 16 Ιανουαρίου 2008 και περιλαμβάνεται BCL, ASP.NET, ADO.NET, τα Windows Forms, WPF, και XML. Ο Scott Guthrie της Microsoft υποσχέθηκε ότι οι βιβλιοθήκες LINQ, WCF, και WF θα προστίθονταν.

Στις 12 Νοεμβρίου 2014, η Microsoft ανακοίνωσε το .NET Core, σε μια προσπάθεια να συμπεριλάβει cross-platform στήριξη για το .NET, την πηγαία έκδοση της εφαρμογής CoreCLR της Microsoft και την υιοθέτηση ενός συμβατικού μοντέλου ανάπτυξης ανοικτού κώδικα υπό την βοηθητική διαχείριση της .NET Foundation. Ο Miguel de Icaza περιγράφει το .NET Core ως "ανασχεδιασμένη έκδοση του .NET που βασίζεται στην απλοποιημένη έκδοση των βιβλιοθηκών κλάσεων ",και ο Immo Landwerth της Microsoft εξήγησε ότι το .NET Core θα είναι "το θεμέλιο όλων των μελλοντικών .NET πλατφόρμων". Κατά τη στιγμή της ανακοίνωσης, η αρχική έκδοση του έργου .NET Core είχε ήδη ένα υποσύνολο του πηγαίου κώδικα των βιβλιοθηκών και συνέπεσε με το relicensing των υφιστάμενων πηγών αναφοράς του .NET της Microsoft μακριά από τους περιορισμούς του Ms-RSL. Ο Landwerth αναγνώρισε τα μειονεκτήματα της προηγούμενης source άδειας που είχε επιλεγεί.

Τον Νοέμβριο του 2014, η Microsoft παρήγαγε επίσης μια ενημέρωση για τις χορηγήσεις των διπλωμάτων ευρεσιτεχνίας της, η οποία εκτείνει περαιτέρω το πεδίο εφαρμογής πέρα από τις προηγούμενες δεσμεύσεις της. Πριν από έργα όπως το Mono υπήρχε σε μια νομική γκρίζα ζώνη. Η νέα πατέντα όμως δεν βάζει όρια στις προδιαγραφές και είναι επεκτάσιμη σε όλες τις τεχνολογίες .NET που περιγράφονται στο MSDN και δεν έχουν καθοριστεί τυπικά από την ECMA, σε περίπτωση που κάποιο έργο επιλέξει να τις υλοποιήσει. Αυτό επιτρέπει στο Mono και σε άλλα έργα

να διατηρήσουν την ισοτιμία χαρακτηριστικών με τα μοντέρνα γνωρίσματα .NET που εισήχθησαν μετά την 4η έκδοση, χωρίς το νομικό κίνδυνο χρήσης πατέντας όσον αφορά την υλοποίηση αυτών των χαρακτηριστικών. Η νέα επιχορήγηση διατηρεί τον περιορισμό ότι κάθε υλοποίηση πρέπει να έχει ελάχιστη συμμόρφωση με τα υποχρεωτικά μέρη της προδιαγραφής CLI.

Στις 31 Μαρτίου του 2016 η Microsoft ανακοίνωσε στη Microsoft Build ότι θα δοθεί εντελώς εκ νέου άδεια του Mono υπό την άδεια MIT, ακόμη και σε περιπτώσεις όπου στο παρελθόν μια εμπορική άδεια ήταν αναγκαία. Η Microsoft συμπληρώνει επίσης προηγούμενη υπόσχεσή ευρεσιτεχνίας για το Mono, δηλώνοντας ότι δεν θα διεκδικήσει οποιαδήποτε εφαρμοσμένη πατέντα εναντίον ομάδων « που χρησιμοποιούν, πωλούν , προσφέρουν προς πώληση, εισάγουν ή διανέμουν το Mono». Ανακοινώθηκε ότι το έργο Mono έχει συμβάλει στο NET Foundation. Οι εξελίξεις αυτές ακολούθησαν την προηγούμενη απόκτηση Xamarin, η οποία ξεκίνησε τον Φεβρουάριο του 2016 και ολοκληρώθηκε στις 18 Μάρτιου 2016.

Το Xamarin επιτρέπει την ανάπτυξη cross-platfrorm εφαρμογών. Οι εφαρμογές μοιράζονται τον ίδιο κώδικα για iOS, Windows, Mac, Android κτλ και παράγει μια ολοκληρωμένη εφαρμογή σε .NET που παρέχει το IDE (monoDevelop IDE και Xamarin plug-in για το Visual Studio). Το Xamarin έχει πάνω από 500.000 προγραμματιστές σε περισσότερες από 120 χώρες σε όλο τον κόσμο.

Το δελτίο Τύπου της Microsoft τονίζει ότι η δέσμευση cross-platform επιτρέπει πλέον μια πλήρως ανοικτή πηγή, σύγχρονο server-side του .NET stack. Ωστόσο, η Microsoft δεν σχεδιάζει να εκδώσει τον πηγαίο κώδικαγια WPF ή Windows.

3.1.1.1 Ιστορία εκδόσεων

Επισκόπηση της ιστορίας εκδόσεων του .NET Framework

Version number	CLR version	Release date	Development tool	Included in Windows	Windows Server	Replaces
1.0	1.0	2002-02-13	Visual Studio .NET[22]	XP[a]	N/A	N/A
1.1	1.1	2003-04-24	Visual Studio .NET 2003[22]	N/A	2003	1.0[23]
2.0	2.0	2005-11-	Visual Studio	N/A	2003, 2003 R2,[25] 2008 SP2,	N/A

		07	2005[24]		2008 R2 SP1	
3.0	2.0	2006-11-06	Expression Blend[26][b]	Vista	2008 SP2, 2008 R2 SP1	2.0[20]
3.5	2.0	2007-11-19	Visual Studio 2008[27]	7, 8[c], 8.1[c], 10[c]	2008 R2 SP1	2.0, 3.0[20]
4.0	4	2010-04-12	Visual Studio 2010[28]	N/A	N/A	N/A
4.5	4	2012-08-15	Visual Studio 2012[29]	8	2012	4.0[20]
4.5.1	4	2013-10-17	Visual Studio 2013[30]	8.1	2012 R2	4.0, 4.5[20]
4.5.2	4	2014-05-05	N/A	N/A	N/A	4.0–4.5.1[20]
4.6	4	2015-07-20	Visual Studio 2015[31]	10	N/A	4.0–4.5.2[20]
4.6.1	4	2015-11-17[32]	Visual Studio 2015 Update 1	10 Version 1511	N/A	4.0–4.6[20]

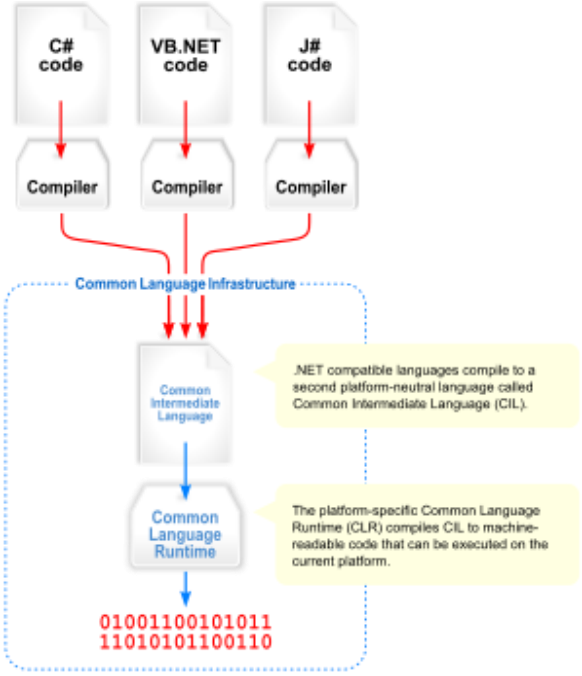
Σημείωση:

a.^ Το .NET Framework 1.0 είναι αναπόσπαστο συστατικό του OS της έκδοσης Windows XP Media Center ή έκδοση Tablet PC. Το CD εγκατάστασης για τις εκδόσεις Home και Professional εκδόσεις των Windows XP SP1, SP2 ή SP3 έρχεται με τα πακέτα εγκατάστασης του .NET Framework.

b.^ Το Expression Blend καλύπτει μόνο το μέρος Windows Presentation Foundation του .NET Framework 3.0.

c.^ ^ ^ Το .NET Framework 3.5 δεν εγκαθίσταται αυτόματα με τα Windows 8, 8,1 ή 10. Θα πρέπει να εγκατασταθεί είτε από ένα μέσο εγκατάστασης των Windows ή από το Internet σε ζήτηση. Ο πίνακας Ελέγχου προσπαθεί πάντα το τελευταίο.

3.2 Διαδικασία Εκτέλεσης CLI



Εικόνα 6: Οπτική επισκόπηση της Common Language Infrastructure (CLI)

3.3 Common Language Infrastructure

Η Common Language Infrastructure (CLI) παρέχει μια πλατφόρμα ουδέτερη γλώσσας για την ανάπτυξη εφαρμογών και εκτέλεση, συμπεριλαμβανομένων των λειτουργιών για συλλογή απορριμάτων, ασφάλεια και διαλειτουργικότητα. Η εφαρμογή των βασικών πτυχών του .NET Framework εμπίπτει στο πεδίο εφαρμογής του CLI. Η λειτουργία δεν συνδέεται με μία μόνο γλώσσα, αλλά είναι διαθέσιμη σε πολλές γλώσσες που υποστηρίζονται από το πλαίσιο. Η εφαρμογή της CLI από τη Microsoft είναι η Common Language Runtime (CLR). Χρησιμεύει ως μηχανή εκτέλεσης του .NET Framework. Όλα τα προγράμματα του .NET εκτελούνται υπό την εποπτεία της CLR, εγγυώμενα ορισμένες ιδιότητες και συμπεριφορές στους τομείς της διαχείρισης της μνήμης, την ασφάλεια και τον χειρισμό εξαιρέσεων.

Προκειμένου τα προγράμματα υπολογιστή να τρέξουν σε CLI, πρέπει να μεταφράζονται σε Common Intermediate Language (CIL) – και όχι σε κώδικα μηχανής. Κατά την εκτέλεση, μια αρχιτεκτονική compiler (JIT) μετατρέπει τον κώδικα CIL σε κώδικα μηχανής. Για τη βελτίωση των επιδόσεων, ωστόσο, το .NET Framework έρχεται με Native Image Generator (NGEN), η οποία εκτελεί compilator σε διαφορετικό χρονισμό.

3.4 Συμβολικές γλώσσες

Ο μεταγλωττισμένος κώδικα CLI είναι αποθηκευμένος σε CLI assemblies. Όπως επιτάσσει η προδιαγραφή, τα assemblies αποθηκεύονται σε μορφή αρχείου Portable Executable (PE), κοινή στην πλατφόρμα των Windows για όλα τα DLL και EXE αρχεία. Κάθε διάταξη αποτελείται από ένα ή περισσότερα αρχεία, ένα από τα οποία πρέπει να περιέχει μεταδεδομένα για τη συμβολική γλώσσα. Το πλήρες όνομα του assembly (δεν πρέπει να συγχέεται με το όνομα του αρχείου στο δίσκο) περιέχει το απλό όνομα σε μορφή κειμένου, τον αριθμό έκδοσης, τη φύση, και το δημόσιο key token. Τα συγκροτήματα θεωρούνται ισοδύναμα αν έχουν το ίδιο πλήρες όνομα.

Ένα ιδιωτικό κλειδί μπορεί επίσης να χρησιμοποιηθεί από τον δημιουργό για ισχυρή ονομασία. Το public key token προσδιορίζει με ποίο ιδιωτικό κλειδί έχει υπογραφεί ένα assembly. Μόνο ο δημιουργός του keypair (συνήθως ο .NET προγραμματιστής υπογράφει τη διάταξη) μπορεί να υπογράψει assemblies που έχουν το ίδιο ισχυρό όνομα με κάποια προηγούμενη έκδοση, δεδομένου ότι ο δημιουργός έχει στην κατοχή του το ιδιωτικό κλειδί. Ισχυρή ονομασία απαιτείται για να προστεθούν assemblies στη Global Assembly Cache.

3.4.1 Βιβλιοθήκη Κλάσεων

Το .NET Framework περιλαμβάνει ένα σύνολο τυποποιημένων βιβλιοθηκών κλάσεων. Η βιβλιοθήκη κλάσεων είναι οργανωμένη σε μια ιεραρχία ονομάτων χώρων (namespaces). Τα περισσότερα από τα built-in API αποτελούν μέρος του System.* ή του Microsoft.* namespace. Αυτές οι βιβλιοθήκες κλάσεων εφαρμόζουν ένα μεγάλο αριθμό κοινών λειτουργιών, όπως την ανάγνωση και γραφή αρχείων, τη γραφική απόδοση, την αλληλεπίδραση βάσεων δεδομένων, καθώς και τη διαχείριση του εγγράφου XML. Οι βιβλιοθήκες κλάσεων του .NET είναι διαθέσιμες για όλες τις συμβατές γλώσσες με CLI. Η βιβλιοθήκη κλάσεων του .NET Framework είναι χωρισμένη σε 2 μέρη (χωρίς σαφή όρια): στη Base Class Library (BCL) και στη Framework Class Library (FCL).

Η BCL περιλαμβάνει ένα μικρό υποσύνολο του συνόλου της βιβλιοθήκης κλάσεων και είναι το βασικό σύνολο των κατηγοριών που χρησιμεύουν ως βασικό API της CLR. Για το .NET Framework οι περισσότερες κλάσεις θεωρούνται κομμάτι της BCL που ανήκουν στο mscorlib.dll, System.dll και στο System.core.dll. Οι κλάσεις BCL είναι διαθέσιμες στο .NET Framework καθώς και οι εναλλακτικές του εφαρμογές,

συμπεριλαμβανομένων των .NET Compact Framework, Microsoft Silverlight, .NET Core και Mono.

Η FCL είναι ένα υπερσύνολο της BCL και αναφέρεται σε ολόκληρη τη βιβλιοθήκη κλάσεων που αποστέλλεται μαζί το.NET Framework. Περιλαμβάνει ένα εκτεταμένο σύνολο από βιβλιοθήκες, συμπεριλαμβανομένων των Windows Forms, ASP.NET και Windows Presentation Foundation (WPF) αλλά επίσης και επεκτάσεις για τις βιβλιοθήκες βασικής τάξης ADO.NET, Language Integrated Query (LINQ), Windows Communication Foundation (WCF), και Workflow Foundation (WF). Η FCL είναι πολύ μεγαλύτερη σε έκταση από τις standard βιβλιοθήκες για τις γλώσσες όπως η C++, και μπορεί να συγκριθεί με τις πρότυπες βιβλιοθήκες της Java.

Με την εισαγωγή των εναλλακτικών εφαρμογών (π.χ. Silverlight), η Microsoft εισήγαγε την έννοια των Portable Class Libraries (PCL), που επιτρέπει σε μια καταναλωτική βιβλιοθήκη να τρέξει σε περισσότερες από μία πλατφόρμες. Με την περαιτέρω εξάπλωση των περαιτέρω .NET πλατφόρμων, η PCL προσέγγιση απέτυχε να αναβαθμιστεί (οι PCLs ορίζεται ως διασταυρώσεις των API μεταξύ δύο ή περισσότερων πλατφόρμων). Ως το επόμενο εξελικτικό βήμα της PCL, η Standard Library .NET δημιουργήθηκε αναδρομικά με βάση τα APIs που βασίζονται στο System.Runtime.dll του UWP και του Silverlight. Οι νέες .NET πλατφόρμες ενθαρρύνονται να εφαρμόσουν μια έκδοση της πρότυπης βιβλιοθήκης που επιτρέπει την επαναχρησιμοποίηση των υφιστάμενων βιβλιοθήκων τρίτων για να τρέξει χωρίς τη νέα έκδοση τους. Το πρότυπο βιβλιοθήκης .NET επιτρέπει την ανεξάρτητη εξέλιξη της βιβλιοθήκης και του μοντέλου της εφαρμογής των στρωμάτων μέσα στην αρχιτεκτονική του .NET.

3.5 Μοντέλα Εφαρμογών

Στην κορυφή των βιβλιοθηκών τάξης, πολλαπλά μοντέλα εφαρμογών χρησιμοποιούνται για τη δημιουργία εφαρμογών. Το .NET Framework υποστηρίζει Console, Windows Forms, Windows Presentation Foundation, ASP.NET και ASP.NET Core εφαρμογές από προεπιλογή. Άλλα μοντέλα εφαρμογών προσφέρονται από εναλλακτικές εφαρμογές του .NET Framework. Το Console, UWP και ASP.NET Core είναι διαθέσιμα στο .NET Core. Το Mono χρησιμοποιείται για την τροφοδοσία των Xamarin μοντέλων εφαρμογών για iOS, Android και OS X. Ο αναδρομικός αρχιτεκτονικός ορισμός των μοντέλων app εμφανίστηκε στις αρχές του 2015 και

εφαρμόστηκε επίσης σε προηγούμενες τεχνολογίες, όπως οι Windows Forms και WPF.

3.5.1 C++/CLI

Η Microsoft παρουσίασε τη C ++ / CLI στο Visual Studio 2005, η οποία είναι μια γλώσσα και ταυτόχρονα μέσο μετάφρασης της Visual C ++ για να τρέξει εντός του .NET Framework. Ορισμένα τμήματα του προγράμματος της C ++ εξακολουθούν να τρέχουν μέσα σε ένα μη διαχειριζόμενο Visual C ++ Runtime, ενώ ειδικά τροποποιημένα τμήματα μεταφράζονται σε κώδικα CIL για να τρέξει με τη CLR του .NET Framework του.

Assemblies που μεταφράζονται χρησιμοποιώντας το μεταγλωττιστή της C ++ / CLI είναι γνωστές ως assemblies μεικτής λειτουργίας, αφού περιέχουν και εσωτερικό και διαχειριζόμενο κώδικα μέσα στο ίδιο αρχείο DLL. Σε τέτοια assemblies είναι επίσης δύσκολο να αναστραφεί η λειτουργία τους, δεδομένου ότι οι .NET decompilers όπως ο.NET Reflector αποκαλύπτουν μόνο το διαχειριζόμενο κώδικα.

3.6 Αρχές σχεδιασμού

3.6.1 Διαλειτουργικότητα

Επειδή τα συστήματα πληροφορικής απαιτούν συνήθως αλληλεπίδραση μεταξύ των νεότερων και των παλαιότερων εφαρμογών, το .NET Framework παρέχει τα μέσα για πρόσβαση στις λειτουργίες που εφαρμόζονται σε νεότερα και παλαιότερα προγράμματα που εκτελούνται έξω από το .NET περιβάλλον. Η πρόσβαση στα στοιχεία COM παρέχεται σε System.Runtime.InteropServices και System.EnterpriseServices χώρους ονομάτων του πλαισίου.

3.6.2 Ανεξαρτησία της γλώσσας

Το .NET Framework εισάγει ένα Common Type System (CTS) που προσδιορίζει όλους του πιθανούς τύπους δεδομένων και δομές προγραμματισμού που υποστηρίζονται από το CLR και πώς μπορούν ή δεν μπορούν να αλληλεπιδρούν μεταξύ τους σύμφωνα με τις προδιαγραφές του CLI. Εξαιτίας αυτής της δυνατότητας, το .NET Framework υποστηρίζει την ανταλλαγή των τύπων και περιπτώσεις

αντικειμένων μεταξύ βιβλιοθηκών και εφαρμογών που έχουν γραφτεί χρησιμοποιώντας οποιοδήποτε γλώσσα που συμμορφώνεται στο .NET.

3.6.3 Ασφάλεια τύπων

Το CTS και το CLR που χρησιμοποιούνται στο .NET Framework ενισχύουν επίσης την ασφάλεια τύπων. Αυτό αποτρέπει τις ασαφείς μετατροπές, τις λανθασμένες μεθόδους κλήσης και τα θέματα του μεγέθους της μνήμης κατά την πρόσβαση σε ένα αντικείμενο. Αυτό καθιστά επίσης τις περισσότερες γλώσσες CLI στατικές (με ή χωρίς εξαγωγή τύπων). Ωστόσο, αρχίζοντας με το .NET Framework 4.0, η Dynamic Language Runtime επέκτεινε τη CLR επιτρέποντας δυναμικά δακτυλογραφημένες γλώσσες να εφαρμοστούν πάνω από τη CLI.

3.6.4 Φορητότητα

Ενώ η Microsoft δεν έχει εφαρμόσει το πλήρες πλαισίο σε οποιοδήποτε σύστημα εκτός των Microsoft Windows, έχει κατασκευάσει το πλαίσιο να είναι ανεξάρτητο από την πλατφόρμα, και εφαρμογές cross-platform είναι διαθέσιμες για άλλα λειτουργικά συστήματα (π.χ. Silverlight). Η Microsoft υπέβαλε τις προδιαγραφές για τη CLI, C#, και C++/CLI και στα δύο, ECMA και ISO, κάνοντας τα διαθέσιμα ως επίσημα πρότυπα. Αυτό καθιστά δυνατή για τρίτους τη δημιουργία συμβατών εφαρμογών πλαισίου και των γλωσσών της σε άλλες πλατφόρμες.

3.6.5 Ασφάλεια

Το .NET Framework έχει το δικό του μηχανισμό ασφαλείας με δύο γενικά χαρακτηριστικά: Τον κωδικό πρόσβασης ασφαλείας (CAS), και την επικύρωση και επαλήθευση. Ο CAS βασίζεται σε στοιχεία που σχετίζονται με μια συγκεκριμένη διάταξη. Συνήθως η απόδειξη είναι η πηγή της διάταξης (αν είναι εγκατεστημένο στον τοπικό υπολογιστή ή έχει κατέβει από το intranet ή το Internet). Ο CAS χρησιμοποιεί στοιχεία για να καθορίσει τα δικαιώματα που χορηγούνται στον κώδικα. Άλλοι κώδικες μπορεί να απαιτούν από τον κώδικα που καλεί να έχει κάποια ειδική άδεια.

3.6.6 Διαχείριση μνήμης

Η CLR ελευθερώνει τον προγραμματιστή από το βάρος της διαχείρισης μνήμης (κατανομή και απελευθέρωση όταν γίνεται), χειρίζεται η ίδια τη διαχείριση της μνήμης από την ανίχνευση όταν η μνήμη μπορεί να απελευθερωθεί με ασφάλεια. Στιγμιότυπα τύπων .NET (αντικείμενα) κατανέμονται από την διαχείριση σωρού(stack), μια περιοχή μνήμης που διαχειρίζεται η CLR. Εφ 'όσον υπάρχει μία αναφορά σε ένα αντικείμενο, το οποίο θα μπορούσε να είναι είτε μια άμεση αναφορά σε ένα αντικείμενο ή μέσω ενός γράφου αντικειμένων, το αντικείμενο θεωρείται ότι είναι σε χρήση. Όταν δεν υπάρχει αναφορά σε ένα αντικείμενο, και αυτό δεν μπορεί να επιτευχθεί ή να χρησιμοποιηθεί, μετατρέπεται σε σκουπίδια (garbage), επιλέξιμα για συλλογή.

Το .NET Framework περιλαμβάνει ένα συλλέκτη σκουπιδιών (GC), το οποίο τρέχει σε τακτά χρονικά διαστήματα, σε ένα ξεχωριστό νήμα από το νήμα της εφαρμογής, που απαριθμεί όλα τα άχρηστα αντικείμενα και διεκδικεί τη μνήμη που τους έχει χορηγηθεί. Ο GC εκτελείται μόνο όταν έχει χρησιμοποιηθεί μία ορισμένη ποσότητα μνήμης ή υπάρχει αρκετή πίεση για τη μνήμη του συστήματος.

Κάθε NET εφαρμογή έχει μια σειρά από ρίζες, οι οποίες είναι δείκτες σε αντικείμενα σχετικά με τη διαχείριση του σωρού (διαχειριζόμενα αντικείμενα). Αυτά περιλαμβάνουν αναφορές σε στατικά αντικείμενα και τα αντικείμενα που ορίζονται ως τοπικές μεταβλητές ή παραμέτρους μεθόδου σήμερα στο πεδίο εφαρμογής. Όταν εκτελείται το GC, σταματά την εφαρμογή και έπειτα, για κάθε αντικείμενο που αναφέρεται στη ρίζα, αναδρομικά απαριθμεί όλα τα αντικείμενα που είναι προσβάσιμα από τα αντικείμενα ρίζας και τα σημαίνει ως προσεγγίσιμα (reachable). Χρησιμοποιεί μεταδεδομένα CLI για να ανακαλύψει ποια αντικείμενα ενθυλακώνονται σε ένα αντικείμενο.

Έπειτα, απαριθμεί όλα τα αντικείμενα στη στοίβα, τα οποία αρχικά ήταν τοποθετημένα με συνεχή τρόπο. Όσα αντικείμενα δεν σημαίνονται ως προσεγγίσιμα αποτελούν σκουπίδια. Αυτή είναι η φάση σήμανσης. Δεδομένου ότι η μνήμη που καταλαμβάνεται από τα σκουπίδια δεν έχει κάποια συνέπεια, θεωρείται ελεύθερος χώρος. Ωστόσο, αυτό αφήνει κομμάτια του ελεύθερου χώρου μεταξύ των αντικειμένων τα οποία ήταν αρχικά συνεχόμενα. Τα αντικείμενα στη συνέχεια συμπιέζονται μαζί για να κάνουν ελεύθερο χώρο για την διαχείριση του συνεχόμενου σωρού και πάλι. Οποιαδήποτε αναφορά σε ένα αντικείμενο αποδυναμώνεται από τη

μετακίνηση του αντικειμένου ενημερώνεται με GC για να αντικατοπτρίζει τη νέα θέση. Η εφαρμογή συνεχίζεται μετά τη συλλογή των απορριμμάτων. Η τελευταία έκδοση του πλαισίου .NET χρησιμοποιεί ταυτόχρονη συλλογή των απορριμμάτων, μαζί με τον κωδικό χρήστη, κάνοντας δυσδιάκριτες παύσεις, γιατί γίνεται στο παρασκήνιο.

Ο GC που χρησιμοποιούνται από το .NET Framework έχει επίσης χαρακτήρα γενεών. Τα πρόσφατα δημιουργημένα αντικείμενα ανήκουν στη γενιά 0. Τα αντικείμενα που επιβιώνουν μια συλλογή απορριμάτων ανήκουν στην γενιά 1 (Generation 1) , και τα αντικείμενα Generation 1 που επιβιώνουν μια άλλη συλλογή είναι Generation 2 αντικείμενα. Το .NET Framework χρησιμοποιεί έως Generation 2 αντικείμενα. Σε αντικείμενα υψηλότερες γενιάς τα σκουπίδια συλλέγονται λιγότερο συχνά από ό, τι στα αντικείμενα κατώτερης γενιάς. Αυτό βοηθά στην αύξηση της αποδοτικότητας της συλλογής σκουπιδιών, καθώς τα μεγαλύτερα αντικείμενα τείνουν να έχουν μεγαλύτερη διάρκεια ζωής από ό, τι τα νεότερα αντικείμενα. Έτσι, με την εξάλειψη των παλαιοτέρων (και επομένως με περισσότερες πιθανότητες να επιβιώσουν από μια συλλογή απορριμάτων) αντικειμένων από το πεδίο εφαρμογής της συλλογής απορριμάτων, λιγότερα αντικείμενα χρειάζεται να ελέγχονται και να συμπιέζονται.

3.6.7 Απλοποιημένη εγκατάσταση

Το NET Framework περιλαμβάνει σχεδιαστικά χαρακτηριστικά και εργαλεία που βοηθούν στη διαχείριση της εγκατάστασης του λογισμικού του υπολογιστή για να βεβαιωθείτε ότι δεν έρχεται σε αντίθεση με το παρελθοντικά εγκατεστημένο λογισμικό, και ότι είναι σύμφωνο με τις απαιτήσεις ασφάλειας.

3.6.8 Απόδοση

Όταν μια εφαρμογή τρέχει για πρώτη φορά, το .NET Framework μεταγλωττίζει τον κώδικα CIL σε εκτελέσιμο κώδικα χρησιμοποιώντας compiler just-in-time, και αποθηκεύει προσωρινά το εκτελέσιμο πρόγραμμα στο .NET Native Cache Image. Λόγω της προσωρινής αποθήκευσης, η εφαρμογή ξεκινά γρηγορότερα για τις επόμενες παρουσιάσεις, αν και η πρώτη εκτέλεση είναι συνήθως πιο αργή. Για να αυξηθεί η ταχύτητα της πρώτης εκκίνησης, οι προγραμματιστές μπορούν να χρησιμοποιήσουν το βοηθητικό πρόγραμμα Native Image Generator.

Ο συλλέκτης απορριμμάτων, που είναι ενοποιημένος με το περιβάλλον, μπορεί να εισάγει μη αναμενόμενες καθυστερήσεις στην εκτέλεση, στις οποίες ο developer δεν μπορεί να έχει άμεσο έλεγχο.

Σε μεγάλες εφαρμογές, ο αριθμός των αντικειμένων που πρέπει να ασχοληθεί ο συλλέκτης απορριμμάτων μπορεί να γίνει πολύ μεγάλος, πράγμα που σημαίνει ότι μπορεί να πάρει πολύ χρόνο για να τα επισκεφθεί και να τα οργανώσει όλα.

Το .NET Framework παρέχει υποστήριξη για την κλήση Streaming Επεκτάσεων SIMD (SSE) μέσω του διαχειρίσιμου κώδικα από τον Απρίλιο του 2004 στο Visual Studio 2013 Update 2. Όμως το Mono παρείχε από πριν υποστήριξη επεκτάσεων SIMD από την έκδοση 2.2 στο χώρο ονομάτων Mono.Simd. Ο βασικός developer του Mono, Miguel de Icaza έχει εκφράσει την ελπίδα ότι η υποστήριξη SIMD θα υιοθετηθεί και από το πρότυπο ECMA του CLR. Οι Streaming Επεκτάσεις SIMD είναι διαθέσιμες στις x86 CPU από την εισαγωγή του Pentium III. Άλλες αρχιτεκτονικές, όπως οι ARM και MIPS έχουν επίσης επεκτάσεις SIMD. Σε περίπτωση που η CPU δεν μπορεί να υποστηρίξει τέτοιες επεκτάσεις, οι εντολές προσομοιώνονται σε επίπεδο λογισμικού.

3.7 Εναλλακτικές υλοποιήσεις

Το πλαίσιο .NET είναι η δεσπόζουσα υλοποίηση των τεχνολογιών .NET. Υπάρχουν και άλλες υλοποιήσεις για τα μέρη του πλαισίου. Αν και η μηχανή εκτέλεσης περιγράφεται από την προδιαγραφή ECMA/ISO, άλλες υλοποιήσεις της μπορεί παρεμποδίζονται από θέματα πατέντας. Τα πρότυπα ISO μπορεί να φέρουν την εξής αποποίηση: «Δίνεται προσοχή στην πιθανότητα κάποια από τα στοιχεία αυτού του εγγράφου να υπόκεινται σε δικαιώματα πατέντας. Το ISO δεν θεωρείται υπεύθυνο για την αναγνώριση μερικών ή όλων τέτοιων δικαιωμάτων». Είναι πιο δύσκολο να αναπτυχθούν εναλλακτικές υλοποιήσεις του FCL, το οποίο δεν περιγράφεται από κάποιο ανοικτό πρότυπο και μπορεί να υπόκειται σε δικαιώματα copyright. Επιπλέον, μέρη του FCL έχουν συγκεκριμένη λειτουργικότητα και συμπεριφορά στα Windows, οπότε η υλοποίηση σε πλατφόρμες πέραν των Windows μπορεί να είναι προβληματική.

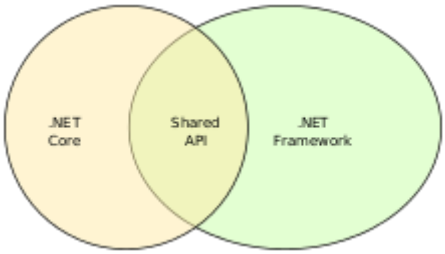
Μερικές εναλλακτικές υλοποιήσεις των τμημάτων του πλαισίου αναφέρονται εδώ.

- Το .NET Micro Framework είναι μια πλατφόρμα .NET για εξαιρετικά περιορισμένους πόρους συσκευών. Περιλαμβάνει μια μικρή έκδοση της CLR και

στηρίζει την ανάπτυξη στην C# (αν και ορισμένοι προγραμματιστές ήταν σε θέση να χρησιμοποιήσουν VB.NET, έστω και με ένα βαθμό πειρατείας, και με περιορισμένες λειτουργίες) και τον εντοπισμό σφαλμάτων (σε εξομοιωτή ή σε υλικό), και χρησιμοποιώντας το Microsoft Visual Studio. Διαθέτει επίσης ένα υποσύνολο του .NET Framework Class Library, ενός πλαισίου GUI και επιπλέον βιβλιοθήκες ειδικά για ενσωματωμένες εφαρμογές.

- Το .NET Core είναι μια εναλλακτική λύση της Microsoft εφαρμογής του πλαισίου διαχειρίσιμου κώδικα, έχει ομοιότητες με το .NET Framework και ακόμη μοιράζεται κάποια API, αλλά έχει σχεδιαστεί με βάση διαφορετικά σύνολα αρχών: Είναι cross-platform, δωρεάν και open-source.
- Το Mono είναι η υλοποίηση των CLI και FCL και παρέχει πρόσθετη λειτουργικότητα. Φέρει διπλή άδεια: τόσο ως δωρεάν λογισμικό όσο και ως αποκλειστικό λογισμικό. Υποστηρίζει ASP.NET, ADO.NET και βιβλιοθήκες Windows Forms για μια ευρεία γκάμα αρχιτεκτονικών και λειτουργικών συστημάτων. Επίσης περιέχει C# και VB.NET compilers.
- Το Portable.NET (κομμάτι του DotGNU) παρέχει μια υλοποίηση του CLI, τμήματα των FCL, και ένα C# compiler. Υποστηρίζει μια ποικιλία συστημάτων επεξεργασίας και λειτουργικό. Το έργο έχει παροπλιστεί, όμως, με την τελευταία σταθερή έκδοση να είναι το 2009.
- Η Shared Source Common Language Infrastructure της Microsoft είναι μια εμπορική υλοποίηση του CLR. Όμως, η τελευταία της έκδοση τρέχει μόνο σε Microsoft Windows XP SP2, και δεν έχει ενημερωθεί από το 2006. Συνεπώς, δεν περιέχει όλα τα χαρακτηριστικά της έκδοσης 2.0. του .NET Framework.
- Το CrossNet αποτελεί μια υλοποίηση του CLI με κομμάτια του FCL. Είναι δωρεάν λογισμικό υπό την ανοικτή άδεια MIT.

3.8 Το .NET Core



Εικόνα 7 :Ένα Venn διάγραμμα που δείχνει τα APIs που καλύπτονται από το .NET Framework, .NET Core and both

Το .NET Core είναι ένα δωρεάν cross-platform πλαίσιο ανοικτού κώδικα, παρόμοιο με το .NET Framework. Αποτελείται από το CoreCLR, μια πλήρης εφαρμογή χρόνου εκτέλεσης cross-platform της CLR, της εικονικής μηχανής που διαχειρίζεται την εκτέλεση των προγραμμάτων NET. Το CoreCLR έρχεται με ένα βελτιωμένο just-in-time compiler, που ονομάζεται RyuJIT. Το .NET Core επίσης περιλαμβάνει το CoreFX, που αποτελεί ένα σκέλος του BCL. Ενώ το .NET Core μοιράζεται ένα υποσύνολο των API του.NET Framework, έρχεται με το δικό του API, που δεν είναι μέρος του .NET Framework. Περαιτέρω, το .NET Core περιέχει το CoreRT. Η διεπαφή command line του .NET Core αποτελεί ένα σημείο εισαγωγής εκτέλεσης για λειτουργικά συστήματα και παρέχει στον developer υπηρεσίες όπως η μεταγλώττιση και η διαχείριση πακέτων.

Το .NET Core υποστηρίζει τέσσερα σενάρια cross-platform: το ASP.NET Coreweb apps, το command-line apps, το libraries, και το Universal Windows Platform apps. Δεν εφαρμόζεται σε Windows Forms ή σε WPF με πρότυπο GUI για την επιφάνεια εργασίας του λογισμικού στα Windows. Το .NET Core είναι επίσης αρθρωτό, πράγμα που σημαίνει ότι, αντί των assemblies, οι προγραμματιστές ασχολούνται με τα πακέτα NuGet. Σε αντίθεση με το .NET Framework, το οποίο εξυπηρετείται χρησιμοποιώντας το Windows Update, το .NET Core βασίζεται στο διαχειριστή πακέτων για την λήψη ενημερώσεων.

Το .NET Core 1.0 κυκλοφόρησε στις 27 Ιουνίου του 2016 μαζί με το Visual Studio 2015 Update 3, που επιτρέπει την ανάπτυξη του .NET Core.

3.9 Τι είναι το ASP .NET

Το ASP.NET είναι ένα open-source, server-side πλαίσιο για web εφαρμογές, σχεδιασμένο για web development αλλά και για την παραγωγή δυναμικών ιστοσελίδων 1 . Αναπτύχθηκε από τη Microsoft για να επιτρέψει στους προγραμματιστές την δημιουργία δυναμικών ιστοσελίδων, web εφαρμογών και υπηρεσιών web.

Κυκλοφόρησε για πρώτη φορά τον Ιανουάριο του 2002 με την έκδοση 1.0 του .NET Framework, και είναι ο διάδοχος της τεχνολογίας Microsoft Active Server Pages (ASP). Το ASP.NET βασίζεται στην Common Language Runtime (CLR), επιτρέποντας στους προγραμματιστές να γράφουν κώδικα ASP.NET χρησιμοποιώντας οποιαδήποτε υποστηριζόμενη γλώσσα .NET. Το πλαίσιο επέκτασης SOAP ASP.NET επιτρέπει στα συστατικά της ASP.NET να επεξεργάζονται τα μηνύματα SOAP.

3.10 Μοντέλα του ASP.NET

Το ASP.NET 4 είναι ένα πλαίσιο ανάπτυξης για την οικοδόμηση ιστοσελίδες και ιστοσελίδες με HTML, CSS, JavaScript και scripting διακομιστή. Υποστηρίζει τρία διαφορετικά μοντέλα ανάπτυξης: Ιστοσελίδες (Web Pages), MVC (Model View Controller), και τις διαδικτυακές φόρμες (Web Forms).

Web Pages	MVC	Web Forms
Παρόμοιο με την PHP και το κλασικό ASP	Model View Controller	Παραδοσιακό ASP.NET

Εικόνα 6: Μοντέλα ανάπτυξης του ASP.NET 4.

Το 2015 η Microsoft κυκλοφόρησε του ASP.NET 5. Το ASP.NET 5 είναι ένα σημαντικός ανασχεδιασμός της ASP.NET2. Το ASP.NET MVC, το ASP.NET Web API, και οι ιστοσελίδες ASP.NET (μια πλατφόρμα η οποία χρησιμοποιεί μόνο σελίδες Razor) έχουν συγσωνευθεί σε ένα ενιαίο MVC 63.

Περιλαμβάνει τα ακόλουθα χαρακτηριστικά:

- Υποστήριξη Linux

[1] https://en.wikipedia.org/wiki/ASP.NET
[2] http://www.w3schools.com/aspnet/
[3] http://docs.asp.net/en/latest/conceptual-overview/aspnet.html#unify

- Υποστήριξη OSX
- Υποστήριξη Node.js
- Υποστήριξη AngularJS
- Tag Helpers
- Components Προβολής
- WEB API
- Υποστήριξη GruntJS
- Υποστήριξη Bower
- Όχι Visual Basic
- Όχι Web Forms

Το ASP.NET βρίσκεται στην διαδικασία της εκ νέου υλοποίησής του ως ένα σύγχρονο και σπονδυλωτό web πλαίσιο, μαζί με άλλα πλαίσια, όπως το Entity Framework. Το νέο πλαίσιο θα χρησιμοποιεί τη νέα open-source πλατφόρμα .NET Compiler (με την κωδική ονομασία "Roslyn") και θα υποστηρίζει όλες τις πλατφόρμες. Το έργο ονομάζεται ASP.NET vNext.

Συμπέρασμα

Στο συγκεκριμένο κεφάλαιο κατανοούμε τη σημαντικότητα της ASP.NET, καθώς περιέχει μια ευρεία γκάμα χαρακτηριστικών τα οποία είναι απαραίτητα στις web εφαρμογές.

Κεφάλαιο 4
Web Forms

Στο κεφάλαιο 4 θα παραθέσουμε κάποια εισαγωγικά στοιχεία για τις web forms, τις προκλήσεις που δημιουργούν οι ASP.NET web forms, τα χαρακτηριστικά τους, για την λειτουργία των Master Pages, για τα ASP.NET Ajax Controls. Επίσης, θα αναφερθούμε στη διαφορά των Web form με το MVC.Τέλος, θα αναλύσουμε πως δημιουργείται μια απλή σελίδα ASP.NET web form στο Visual studio 2013.

4.1 Introduction to Web Forms

Αυτή η ενότητα, περιγράφει και αναλύει την πλατφόρμα ASP.NET, η οποία περιλαμβάνει τις βασικές τεχνολογίες που χρειάζεται να γνωρίζουν οι σχεδιαστές/ δημιουργοί για να δημιουργήσουν επιτυχημένες διαδικτυακές εφαρμογές. Αρχικά, παρουσιάζεται το περιβάλλον εργασίας και τα βασικά εργαλεία, ώστε να δημιουργηθούν τα προϊόντα της ASP.NET. Κατόπιν ακολουθούν οι υπόλοιπες εφαρμογές. Επιπρόσθετα, περιγράφεται ένα ενδιαφέρον δωρεάν εργαλείο της Microsoft, Web Platform Installer η οποία είναι απαραίτητη για την εγκατάσταση του Visual Studio. Έπειτα παρουσιάζονται οι ικανότητες του Visual Studio και ο τρόπος με τον οποίο δημιουργούνται τα στοιχεία της ASP.NET. Στόχος αυτής της ενότητας είναι αρχικά η δημιουργία μίας απλής διαδικτυακής εφαρμογής και ο τελικός στόχος, μέσω ενασχόλησης με ειδικότερα ζητήματα, η δημιουργία δυναμικών διαδικτυακών εφαρμογών και η εύρεση δεδομένων από διαφορετικές πηγές.

Σε δεύτερο επίπεδο, παρουσιάζονται οι προσφερόμενες δυνατότητες του περιβάλλοντος NET (Net Framework), το οποίο αποτελεί το πρώτο βήμα για την εγκατάσταση της Web Platform Installer. Πρόκειται για μία τεράστια υπολογιστική πλατφόρμα, στην οποία μπορεί να γραφτεί και να εκτελεστεί ένας κώδικας. Αν και ακούγεται απλοϊκό, στην πραγματικότητα είναι μία τεράστια βιβλιοθήκη λογισμικού με ασφαλές περιβάλλον εκτέλεσης, που διαθέτει και διαχειρίζεται ικανότητα ‘μνήμης’. Η πλατφόρμα διαθέτει τη βασική γλώσσα σχεδιασμού (CLR), ώστε να δημιουργεί τον κώδικα και αναγνωρίζει πληθώρα παραγόμενων γλωσσών, καθώς υπάρχουν περίπου 25 διαφορετικές γλώσσες, οι οποίες χρησιμοποιούνται για να δημιουργηθούν πολλές και διαφορετικές εφαρμογές. Αυτό το οποίο κάνει την .NET, τόσο ανταγωνιστική και παραγωγική είναι το Visual Studio, στο οποίο μπορεί να

γραφτεί οποιοσδήποτε κώδικας και να δημιουργηθούν τα απαιτούμενα προϊόντα ευκολότερα από οποιοδήποτε άλλο λογισμικό.

Στο επόμενο επίπεδο μελέτης, εγκαθίσταται μία εκδοχή του Visual Studio με τη χρήση της Web Platform Installer. Η Web Platform Installer μπορεί να ‘κατεβεί΄ από οποιαδήποτε διαδικτυακή μηχανή αναζήτησης, πληκτρολογώντας το όνομά της (Web Platform Installer) και πατώντας το πράσινο πλήκτρο στην ιστοσελίδα της. Είναι μόνο 2 MB, οπότε κατεβαίνει γρήγορα και η εγκατάσταση είναι πολύ εύκολη. Το πλεονέκτημα αυτού του λογισμικού είναι ότι δε χρειάζεται οι χρήστες να αναζητήσουν τα απαιτούμενα εργαλεία χρήσης του σε άλλους τόπους. Επίσης, αυτό το εργαλείο συνδέει αυτόματα εξαρτημένα προγράμματα και βρίσκει συμπληρωματικά προϊόντα ανοικτού κώδικα που βρίσκονται στον ιστοχώρο. Μετά την εγκατάσταση της Web Platform Installer, ακολουθεί το λογισμικό Visual Web Developer Express 2010 SP1, το οποίο είναι δωρεάν εκδοχή του Visual Studio. Επιπρόσθετα υπάρχει η δυνατότητα να προστεθούν και να ‘κατέβουν΄ πάρα πολλά εργαλεία και προϊόντα, σε συνάρτηση με την Web Platform Installer, πατώντας το πλήκτρο της ‘προσθήκης’, δεξιά της κάθε εφαρμογής. Σε περίπτωση μη εύρεσης κάποιας εφαρμογής, πληκτρολογείται τμήμα της ονομασίας της στην τοπική αναζήτηση. Επιπρόσθετα, υπάρχει η δυνατότητα παρακολούθησης των διαδικασιών που πραγματοποιούνται στο παρασκήνιο, ‘πατώντας’ πάνω στα αντικείμενα προς εγκατάσταση. Κατά τη διάρκεια της εγκατάστασης θα ζητηθεί η δημιουργία ενός κωδικού εισόδου στο πρόγραμμα, σε περίπτωση επιθυμίας του χρήστη. Όταν ολοκληρωθεί η διαδικασία εγκατάστασης, το πλήκτρο ‘προσθήκης΄ έχει αντικατασταθεί με το πλήκτρο ‘έγκατεστημένο΄.

Επίσης περιγράφονται τα χαρακτηριστικά του Visual Web Developer Express 2010 SP1 και ο τρόπος χρήσης του, ώστε να δημιουργούνται τα προϊόντα του ASP.NET. Η πλατφόρμα εργασίας του Visual Web Developer Express 2010 SP1 έχει πολλά παράθυρα εργασίας, γραμμές εργαλείων και μενού που χρησιμοποιούνται για να ‘χτιστούν’, σχεδιαστούν, ελεγχθούν και αναπτυχθούν οι εφαρμογές. Ακολουθεί ένα παράδειγμα, ώστε να γίνουν κατανοητές οι δυνατότητες του προγράμματος, όπου περιγράφεται η πλατφόρμα από την αρχική της σελίδα και τα απαιτούμενα βήματα ώστε να ολοκληρωθεί η διαδικασία. Εν τέλει, πραγματοποιείται έλεγχος του παραγόμενου αποτελέσματος στον ιστοχώρο, από προεπιλογή του λογισμικού του ASP.NET. Ένα από τα σημαντικά χαρακτηριστικά του προγράμματος είναι η δυνατότητα οπτικού σχεδιασμού του προϊόντος.

Οι τρόποι που μπορούμε να δημιουργήσουμε διαδικτυακές εφαρμογές με το ASP.NET χωρίζονται σε 2 βασικές κατηγορίες:

α. Web Site Project και

β. Web Application Project.

Το πρώτο επιτρέπει στο χρήστη να δουλεύει απευθείας με φακέλους μέσω του Visual Studio, ενώ το δεύτερο χρησιμοποιείται για τη δημιουργία μίας δυναμικής κατασκευής για το διαδικτυακό προϊόν. Υπάρχουν δύο διαφορετικοί τρόποι για τη δημιουργία των προϊόντων, διαδικασία που περιγράφεται από τον ομιλητή αναλυτικά και συγκεκριμένα είτε μέσω της Δημιουργίας νέου ΄Εργου' (New Project), είτε 'Ιστοσελίδας' (New Website), στην αρχική πλατφόρμα του προγράμματος. Υπάρχουν αρκετές εναλλακτικές ως προς τον τρόπο εργασίας.

Το πρώτο παράδειγμα είναι η δημιουργία μίας νέας ιστοσελίδας από προεπιλογή, όπου περογράφεται με αναλυτικά βήματα η διαδικασία μέχρι το στάδιο της δημοσίευσής της. Ακολουθεί δεύτερο παράδειγμα, όπου δημιουργείται νέο έργο και επεξηγείται εκ νέου η συγκεκριμένη διαδικασία, η οποία είναι διαφορετική από την προηγούμενη. Συγκεκριμένα, η γλώσσα η οποία χρησιμοποιείται για να παραχθεί το προϊόν είναι κωδικοποιημένη, οπότε κάποια στιγμή θα χρειαστεί να αποκωδικοποιηθεί, να μεταγλωττιστεί. Σημείο έναρξης είναι ο πηγαίος κώδικας (source code) της εφαρμογής, ο οποίος πρέπει να περάσει μέσα από το αντίστοιχο πρόγραμμα 'μεταγλώττισης' (language compiler). Το αποτέλεσμα αυτής της διαδικασίας ονομάζεται Microsoft Intermedia Language (MSIL). Η γλώσσα αυτή ορίζεται από τον κώδικα και συγκεκριμένα μεταδεδομένα. Όταν ολοκληρωθεί αυτή η μεταγλώττιση, κατά την εκτέλεση του τελικού προϊόντος προκύπτει μία εγγενής γλώσσα η οποία διατηρείται, ακόμα και αν ο δημιουργός δεν επέμβει περαιτέρω στο προϊόν. Συμπερασματικά, η διαφορά κατά τη δημιουργία μίας ιστοσελίδας και ενός έργου έγκειται στο γεγονός ότι στο έργο είναι απαραίτητη η μεταγλώττιση της γλώσσας που χρησιμοποιείται μέχρι τη δημοσίευση.

Τέλος, παρουσιάζεται η δημιουργία ενός νέου Έργου εκ του μηδενός, χωρίς φόρμα προεπιλογής, ώστε να γίνουν κατανοητές οι διαφορετικές διαδικτυακές μορφές που υπάρχουν προς επεξεργασία και τα διαφορετικά αποτελέσματα που μπορούν να παράγουν. Η φόρμα έναρξης είναι απλή και άδεια, έχοντας μόνο ένα φάκελο διαμόρφωσης. Κατόπιν, ακολουθούν ακόμα μία φορά αναλυτικά τα βήματα δημιουργίας του έργου.

Συνοπτικά μπορούμε να πούμε ότι, οι Web Forms της ASP.NET:

- Βασίζονται στην τεχνολογία της Microsoft ASP.NET, στην οποία ο κώδικας που τρέχει στον server παράγει δυναμικά έξοδο ιστοσελίδας στον browser ή τη συσκευή του client.
- Είναι συμβατές με οποιοδήποτε πρόγραμμα περιήγησης ή κινητή συσκευή. Μια ιστοσελίδα της ASP.NET καθιστά αυτόματα τη σωστή HTML (συμβατή με τον εκάστοτε browser) για χαρακτηριστικά, όπως το στυλ, τη διάταξη, και ούτω καθεξής.
- Είναι συμβατές με κάθε γλώσσα που υποστηρίζεται από το .NET CLR, όπως η Microsoft Visual Basic και Microsoft Visual C #
- Είναι δομημένες πάνω στο πλαίσιο της Microsoft .NET. Αυτό παρέχει όλα τα πλεονεκτήματα του πλαισίου, συμπεριλαμβανομένου ενός διαχειριζομένου περιβάλλοντος, της ασφάλειας συγγραφής και της κληρονομικότητας.
- Είναι ευέλικτες επειδή μπορεί να προστεθούν επιπλέον controls, δημιουργημένα από το χρήστη ή από τρίτους.

Παράλληλα, οι Web Forms του ASP.NET παρέχουν:

- Διαχωρισμός της HTML και άλλων κωδίκων UI από την λογική της εφαρμογής.
- Μια σειρά ρυθμίσεων του εξυπηρετητή της ιστοσελίδας για κοινές εργασίες (πχ τη πρόσβαση στα δεδομένα).
- Ισχυρό data binding, με μεγάλη υποστήριξη εργαλείων.
- Υποστήριξη για client-side scripting που εκτελείται στο πρόγραμμα περιήγησης.
- Υποστήριξη για μια ποικιλία άλλων δυνατοτήτων, συμπεριλαμβανομένης της δρομολόγησης, της ασφάλειας, τις επιδόσεις, τη διεθνοποίηση, τον έλεγχο, τον εντοπισμό σφαλμάτων και την αντιμετώπιση των λαθών.

Συμπερασματικά, η ASP.NET παρέχει μία ικανή πλατφόρμα δημιουργίας εφαρμογών, διαδικασία που υποστηρίζεται από το Web Platform Installer και το Visual Studio παρέχει τον τρόπο ώστε να πραγματοποιούνται οπτικά ο σχεδιασμός και η κωδικοποίηση των εφαρμογών και τέλος η δημιουργία ιστοσελίδας και διαδικτυακής εφαρμογής, είναι διαδικασίες αλληλένδετες και αλληλοϋποστηριζόμενες.

4.2 ASP.NET Web Forms Features

Σε αυτήν την ενότητα, αναφέρονται τα βασικά χαρακτηριστικά των διαδικτυακών μορφών της ASP.NET, ο τρόπος με τον οποίο είναι οργανωμένες και οι διαδικασίες που πραγματοποιούνται στο παρασκήνιο. Σε επόμενο στάδιο εξηγείται η σημασία του code separation ώστε να παραμένει το πρόγραμμα ‘καθαρό’ και λειτουργικό. Επίσης περιγράφεται η λειτουργία της Page Directive, πραγματοποιείται μία εισαγωγή στον Έλεγχο Διαδικτύου (Web Control) και στον τρόπο με τον οποίο ο χρήστης επικυρώνει τις εισαγωγές του στο πρόγραμμα. Επίσης εξηγείται η διαδικασία με την οποία οι χρήστες δημιουργούν περιβάλλοντα από προεπιλογή. Τέλος, περιγράφονται οι ‘Στοιχεία-Βιβλιοθήκες Χρήστη’ (User Controls) και ο τρόπος που δημιουργούνται και χρησιμοποιούνται, ώστε να ελέγχονται οι κεφαλίδες, τα υποσέλιδα και ο,τιδήποτε άλλο είναι εμφανές στις ιστοσελίδες.

Οι διαδικτυακές μορφές της ASP.NET διαθέτουν ορισμένα χαρακτηριστικά που επιτρέπουν στους χρήστες να αυξάνουν την παραγωγικότητά τους και να μειώνουν τη χρονοβόρα διαδικασία γραφής ενός κώδικα, όπως επίσης να περιορίζουν την εμφάνιση ιών. Η ASP.NET παρέχει μία ικανή προς προγραμματισμό ιστοσελίδα, η οποία ονομάζεται Διαδικτυακή Φόρμα. Στόχος της είναι να υποστηρίζει αρχεία HTML με δυναμικό τρόπο σε όλα τα προγράμματα περιήγησης και είναι προσανατολισμένη στο αντικείμενο. Χαρακτηριστικό της είναι ότι πολλές δραστηριότητες πραγματοποιούνται παρασκηνιακά χωρίς να απαιτείται η δημιουργία πολλών κωδίκων από τους χρήστες. Επίσης, κάθε σελίδα που παράγεται προκύπτει από το ίδιο το πρόγραμμα (ASP.NET Page class), γεγονός που βελτιώνει τη λειτουργικότητα. Έτσι, με τη χρήση της Page class, ο χρήστης έχει τη δυνατότητα να παρουσιάζει, συγκεντρώνει, επικυρώνει και ελέγχει τα δεδομένα. Τα χαρακτηριστικά του διαδικτυακών μορφών της ASP.NET χωρίζονται σε ορισμένες κατηγορίες. Η πρώτη κατηγοριοποίηση περιλαμβάνει μορφές ελέγχου στην ασφάλεια, στα δεδομένα, στην πλοήγηση και στα διαδικτυακά τμήματα. Ο έλεγχος ασφάλειας περιλαμβάνει δραστηριότητες όπως η είσοδος και έξοδος σε σελίδες, καταχώρηση χρηστών και υπενθύμιση κωδικών εισόδου. Ο έλεγχος δεδομένων βρίσκει πολλές εφαρμογές σε εταιρείες οι οποίες λαμβάνουν πολλά δεδομένα και με τον τρόπο αυτό έχουν τη δυνατότητα να ταξινομούν τα ουσιώδη από τα ανεπιθύμητα και ο έλεγχος πλοήγησης επιτρέπει τη δημιουργία ιστορικού και σχέσεων ανάμεσα στους τόπους πλοήγησης.

Ορισμένα από τα βασικά χαρακτηριστικά περιλαμβάνουν τη βασική μορφοποιημένη σελίδα (master pages), τα θέματα/υφές που ο δημιουργός μπορεί να επεξεργαστεί, την τοπική/γλωσσική προσαρμογή ανάλογα με τον εκάστοτε χρήστη και την ικανότητα προσαρμογής του παραγόμενου προϊόντος με βάση τον αντίστοιχο πρόγραμμα περιήγησης. Επιπρόσθετα, υπάρχουν ορισμένες παροχές που προσφέρει η εφαρμογή ως προς την ασφάλεια και συγκεκριμένα περιλαμβάνουν έννοιες όπως ιδιότητα μέλους, η διαχείριση ρόλων του χρήστη, η πλοήγηση στον ιστοχώρο, η κρυφή μνήμη και η διαχείριση. Η διαδικτυακή φόρμα περιλαμβάνει αρκετά ‘αντικείμενα’ και συγκεκριμένα οδηγίες (directives) που αφορούν για παράδειγμα τη γλώσσα (π.χ. <%@ Page Language=’’C#’’ AutoEventWireup=’’True’’%>),παροχές ελέγχου (πχ. <asp:Label id=’’lblHelooWorld’’ runat-‘’server’’ />), εντολές χρήστη (π.χ. <acme:Header id=’’ucHeader runat=’’server’’ />), εκφράσεις ASP.NET (π.χ. <%$ ConnectionStrings: NorthwindConnString %>), εκφράσεις σύνδεσης δεδομένων (π.χ. <%# Eval(‘’DBFieldName’’) %>) και πολλά άλλα.

Ένα από τα βασικότερα χαρακτηριστικά της ASP.NET είναι η δυνατότητα διαχωρισμού των αρχείων HTML από τον κώδικα (code) προγραμματισμού της εφαρμογής, διαδικασία που δημιουργεί ‘καθαρότερα’, ευέλικτα και καλύτερα οργανωμένα αποτελέσματα. Εντούτοις, το πρόγραμμα παρέχει τη δυνατότητα δημιουργίας ενιαίου αρχείου HTML και κώδικα, σε περίπτωση που το επιθυμεί ο δημιουργός.

Στην επόμενη ενότητα αναλύονται τα χαρακτηριστικά της Page Directive, η οποία τοποθετείται στην κορυφή της εκάστοτε ASP.NET σελίδας (<%@ Page Language=’’C#’’ %>). Κάποια από αυτά είναι ο καθορισμός της γλώσσας που χρησιμοποιείται στη σελίδα, η διατήρηση των γραμμών εργαλείων, η αναγνώριση του ίχνους του κώδικα και των αρχείων, αναγνώριση βασικού θέματος ή φόρμας σελίδας ή μίας λανθασμένης σελίδας, καθώς επίσης και πολλά άλλα.

Στη συνέχεια ακολουθεί μία περιορισμένη λίστα με τα διάφορα γνωρίσματα οδηγιών για τη σελίδα, η οποία έχει την εξής μορφή: <%@ Page attribute=’’value’’ %>, όπου ‘value’ τοποθετείται κάποιο από τα παρακάτω γνωρίσματα.

Async: όταν η παραγόμενη σελίδα προκύπτει από IHttpAsyncHandler, το οποίο προσδίδει ασύγχρονες ικανότητες

CodeFile: προσδιορίζει το όνομα του φακέλου σε συνάρτηση με τον κώδικα δημιουργίας για την προς δημιουργία σελίδα

EnableTheming: υποδηλώνει εάν συγκεκριμένα θέματα θα εφαρμοστούν στη σελίδα

Language: οριοθετεί τη γλώσσα που χρησιμοποιείται στη σελίδα (C# ή VB)

Trace: Επιτρέπει/ απαγορεύει την λειτουργία ίχνους της σελίδας

MaintainScrollPositionOnPostback: το JavaScript θα ενταχθεί στην παραγόμενη σελίδα, διατηρώντας τη θέση του στον περιηγητή

Theme: προσδιορίζει το όνομα του θέματος που θα χρησιμοποιηθεί στη σελίδα.

Ακολουθεί η παρουσίαση των standard προγραμματιζόμενων κλάσεων για την ανάπτυξη διαδικτυακών εφαρμογών (Web Controls), τα οποία στην πραγματικότητα στοχεύουν στη δυναμική δημιουργία αρχείων HTML. Η ASP.NET στηρίζεται στο Web Control, το οποίο είναι μία ομοταξία με ιδιοκτησία, μεθόδους και γεγονότα, ώστε να συγκεντρώνει, εμφανίζει και αξιολογεί τα προσλαμβάνοντα δεδομένα. Υπάρχουν τέσσερις κατηγορίες ρυθμίσεων, με κοινότερες τις Ρυθμίσεις Διαδικτυακού Εξυπηρετητή (Web Server Controls), οι οποίες πραγματοποιούνται σε υπό προγραμματισμό αντικείμενα, τις Ρυθμίσεις Εξυπηρετητή HTML (HTML Server Controls), τις Ρυθμίσεις Επικύρωσης (Validation Controls), ώστε να επιβεβαιώνονται οι εισαγωγές και τέλος, τις Ρυθμίσεις Χρήστη (User Controls) που περιλαμβάνουν ελέγχους σε επικεφαλίδες, υποσέλιδα και διαμόρφωση της ιστοσελίδας. Στη συνέχεια, ακολουθούν ορισμένα παραδείγματα ρυθμίσεων. Περιληπτικά αναφέρονται τα εξής:

Toolbox-> Standard-> Pointer/AdRotator/BulletedList/Button/Calendar….

Toolbox-> Panel/PlaceHolder/RadioButton/RadioButtonList/Substitution…

Toolbox-> Pointer/AccessDataSource/Chart/DataList/DataPager….

Toolbox->Validation-> Pointer/CompareValidator/CustomValidator…

Toolbox->Navigation-> Pointer/Menu/SiteMapPath/Treeview…

Toolbox->Login-> Pointer/ChangePassword/CreateUserWizard/Login…

Toolbox->AJAX Extensions-> Pointer/ScriptManager/ ScriptManagerProxy…

Επιπρόσθετα τονίζεται ότι η δήλωση του Ελέγχου Εξυπηρετητή στη Διαδικτυακή φόρμα πραγματοποιείται με την εκ των προτέρων τοποθέτηση του ακρωνυμίου ‘asp’ πριν από το όνομα της εντολής. Επίσης, οι ιδιότητες του Ελέγχου Εξυπηρετητή μπορούν να δηλωθούν με τη χρήση και γραφή γνωρισμάτων.

4.2.1 Validation Controls

Επιπρόσθετα, μία εργασία πολύ κοινή που απαιτείται στην ASP.NET είναι η δυνατότητα επικύρωσης των καταχωρήσεων, τόσο από την πλευρά των δημιουργών όσο και των χρηστών, διαδικασία που πραγματοποιείται εύκολα. Υπάρχουν αρκετοί τύποι επικύρωσης, όπως φόρμα απαίτησης εισόδου, επικύρωση συγκεκριμένων κριτηρίων, σύγκριση αριθμών και αξιών, συμβόλων, δεδομένων, σχεδίων και πολλοί ακόμα και η καταχώρησή τους γίνεται ακολουθώντας κάποια από τις επόμενες φόρμες:<asp:RequiredFieldValidator/ RangeValidator/ CompareValidator/ RegularExpressionValidator/ CustomValidator/ ValidationSummary>.

Στο επόμενο στάδιο, παρουσιάζεται ο τρόπος με τον οποίο ο δημιουργός μπορεί να δημιουργήσει κάποιες προεπιλεγμένες διατάξεις στις ASP.NET Web Forms Applications. Στην πρώτη περίπτωση που μπορεί να εφαρμοστεί η προεπιλεγμένη διάταξη είναι στο να δημιουργηθεί ένα πλήκτρο από προεπιλογή, όταν ο χρήστης χρησιμοποιεί το πλήκτρο ‘enter’. Ο τρόπος με τον οποίο ορίζεται μία οδηγία με προεπιλογή είναι όταν χρησιμοποιείται η οδηγία DefaultFocus: <form defaultFocu=’’txtName’’ runat=’’server’’>. Η υποστήριξη προγραμματισμού για την ενεργοποίηση των ενοτήτων είναι: α. Page.SetFocus(control), β. Page.SetFocus(‘’ClientID’’) και γ. txtName.Focus(). Επίσης ακόμα μία εφαρμογή αυτής της διαδικασίας είναι όταν ο δημιουργός ορίζει τα επίπεδα εστίασης στην εκάστοτε σελίδα.

Στην τελευταία ενότητα αυτού του κεφαλαίου, περιγράφεται ο τρόπος με τον οποίο ο δημιουργός θα σχεδιάσει ρυθμίσεις για τον χρήστη (User controls) και πως θα εφαρμοστούν στις παραγόμενες σελίδες. Οι συγκεκριμένες ρυθμίσεις επιτρέπουν σε κοινές διεπιφάνειες χρηστών να κατοχυρωθούν λειτουργικά και να επαναχρησιμοποιηθούν σε διάφορες ιστοσελίδες. Παραδείγματα τέτοιου τύπου είναι οι οθόνες εισόδου που χρησιμοποιούνται σε πολλαπλούς ιστότοπους, οι κεφαλίδες και τα υποσέλιδα, τα επαναλαμβανόμενα μενού και διάφορα άλλα. Ο τρόπος με τον οποίο δημιουργείται αυτός ο τύπος ρύθμισης είναι στην πραγματικότητα μία σελίδα στην οποία έχει προστεθεί μία συγκεκριμένη εντολή (<%@ Control Language=’’C#’’%>) και πρέπει να σώζεται με την επέκταση ‘’.ascx’’.

Συμπερασματικά, σε αυτήν την ενότητα παρουσιάστηκαν κάποια από τα βασικά χαρακτηριστικά του περιβάλλοντος εργασίας της ASP.NET και δόθηκε έμφαση σε ορισμένες από τις οδηγίες σχεδιασμού και τα γνωρίσματά τους. Βασικό τμήμα του

κειμένου ήταν οι οδηγίες διαμόρφωσης σελίδας (page directive) οι οποίες επιτρέπουν στο δημιουργό να διαμορφώνει τις σελίδες χωρίς τη διαδικασία της κωδικοποίησης, καθώς επίσης και μορφές/οδηγίες ρύθμισης (control directive). Στη συνέχεια περιγράφηκε ο τρόπος με τον οποίο ένα αρχείο html δημιουργείται και 'ανεβαίνει' στους οδηγούς περιήγησης, χωρίς μεγάλο βαθμό κωδικοποίησης εκ μέρους του δημιουργού. Τέλος, περιφράφηκαν τα τμήματα της σχεδίασης που δημιουργούνται από προεπιλογή και ο τρόπος με τον οποίο οι ρυθμίσεις των χρηστών μπορούν να επαναχρησιμοποιηθούν και να μοιραστούν εκ νέου.

4.3 Master Pages

Αυτή η ενότητα αναφέρεται στα χαρακτηριστικά των Βασικών/κύριων σελίδων (Master pages) και στον τρόπο με τον οποίο μπορούν να προσφέρουν συνέπεια μορφοποίησης στις εφαρμογές της ASP.NET. Στο πρώτο τμήμα περιγράφεται η ταυτότητα μίας Master page και ο τρόπος με τον οποίο μπορεί να τη χρησιμοποιήσει ο σχεδιαστής. Στη συνέχεια πραγματοποιείται μία παρουσίαση του τρόπου δημιουργίας της, της χρήσης της σε σελίδες περιεχομένων στοιχείων (Content pages) και τον τρόπο με τον οποίο ο δημιουργός μπορεί να επεμβαίνει και να αλλάζει τις Master pages με τρόπο δυναμικό, όπως για παράδειγμα στην περίπτωση που υπάρχει ανάγκη εκτύπωσης τμήματος της σελίδας.

Οι Master pages παρέχουν τον τρόπο, ώστε να διαμορφώνεται μία ενιαία κοινή ρύθμιση σελίδας σε πολλές ιστοσελίδες, σε μία διαδικτυακή εφαρμογή. Επί παραδείγματι, σε μία ιστοσελίδα, η επικεφαλίδα, το μενού και το πλαίσιο κειμένου παραμένουν τα ίδια, ακόμα και όταν αλλάζουμε τις καρτέλες εισόδου, πληροφορικών, επικοινωνίας, παραγγελιών κτλ., με αποτέλεσμα η διάταξη να παραμένει ενιαία. Αυτή η ενιαία ρύθμιση σελίδας, όπως προαναφέρθηκε, πραγματοποιείται χρησιμοποιώντας ένα μόνο φάκελο. Αυτός ο φάκελος, από προεπιλογή, έχει κάποια συγκεκριμένα χαρακτηριστικά τα οποία κληρονομεί από το User Control, καθώς στην πραγματικότητα οι Master pages είναι ένας τύπος του User Control. Ένα ακόμα χαρακτηριστικό αυτών των σελίδων είναι το γεγονός ότι μπορούν να χρησιμοποιηθούν ως εκκολαπτήρια παραγόμενων ιστοσελίδων που αναπτύσσονται κάτω από την ομπρέλα της κεντρικής. Επίσης, καθορίζουν τις περιοχές στις οποίες πρέπει να τοποθετηθούν τα περιεχόμενα της εκάστοτε σελίδας,

εντός της προκαθορισμένης διάταξής της. Παρασκηνιακά μπορούν να έχουν κώδικα γραφής, όπως οι ιστοσελίδες διαφήμισης και έχουν την κατάληξη ‘.master'.

Οι Master pages μπορούν να δημιουργηθούν άμεσα στο Visual Web Express ή Visual Studio με την προσθήκη ενός αντικειμένου του τύπου της Master page στο εκάστοτε μελετώμενο έργο. Αρχικά, παρουσιάζονται τα στοιχεία που βρίσκονται μέσα σε μία Master page. Οι σελίδες αυτές επιτρέπουν στο σύνολο των διατάξεων των σελίδων να ορίζονται σε ένα μόνο σημείο. Συγκεκριμένα, υπάρχει η οδηγία <%@Master%>, η οποία χαρακτηρίζει την εκάστοτε Master page και ακολουθούν όλα εκείνα τα στοιχεία που είναι απαραίτητα για τη διαμόρφωσή της και την τελική της διάταξη στον ιστοχώρο. Επιπρόσθετα στοιχεία είναι το αρχείο Html, οι επικεφαλίδες, υποσέλιδα κτλ. Απαραίτητο είναι το σημείο τοποθέτησης του περιεχομένου των σελίδων, για παράδειγμα <asp:ContentPlaceHolder id=’’ph1’’runat=’’Server’’/>. Στο σημείο αυτό είναι απαραίτητο να τονιστεί ότι οι Master pages έχουν το κείμενο xmlns, τις κεφαλίδες (head) και τον κορμό (body) στη δομή τους, οπότε δεν τοποθετούνται εκ νέου από το δημιουργό, παρά μόνο συμπληρώνονται.

Στο επόμενο στάδιο περιγράφεται ο τρόπος δημιουργίας των Content pages και της συνεργασίας τους με τις Master pages. Οι σελίδες αυτές πάντα έχουν ως σημείο αναφοράς μία Master page και περιέχουν μία ή περισσότερες ρυθμίσεις/εντολές διάταξης. Συγκεκριμένα, υπάρχει η οδηγία <%@Page%> σε συνδυασμό με ένα ακόμα χαρακτηριστικό MastePageFile=’’Mysite.master’’%>, το οποίο παραπέμπει στη διάταξη και περιεχόμενο της σχετιζόμενης Master Page. Απαραίτητη είναι επίσης η εξάρτηση του σημείου τοποθέτησης του περιεχομένου των νέων σελίδων, για παράδειγμα <asp:contented=’’bodycontent’’ ContentPlaceHolderID =’’ph1’’runat=’’Server’’/>. Από αυτό το σημείο και έπειτα, όποιο περιεχόμενο τοποθετείται ανάμεσα στην αρχή και την έξοδο αυτής της ρύθμισης, τοποθετείται στην κατάλληλη θέση, με βάση τη διάταξη της Master page. Κατά τ’ άλλα, η Content page έχει τα γνωστά χαρακτηριστικά, την κατάληξη .aspx, τον κώδικά γραφής τους και όλες τις δυνατότητες του Server Control. Μπορούν να ενταχθούν στις ιστοσελίδες με διάφορους τρόπους.

Οι Master pages έχουν τη δυνατότητα να αλλάζουν με τρόπο δυναμικό, χρησιμοποιώντας το πρόγραμμα Page_Prelnit. Σε αυτό το σημείο περιγράφεται ο σχεδιασμός μίας ικανής για εκτύπωση Master page, όπου στην πραγματικότητα ο χρήστης θα χρησιμοποιεί ένα σύνδεσμο εκτύπωσης, ο οποίος θα ενεργοποιεί τη Master page, ώστε να εκτυπώνει μόνο το περιεχόμενο της σελίδας. Για να

πραγματοποιηθεί αυτή η διαδικασία πρέπει να αλλάξουν οι εντολές για τις διατάξεις της αρχικής σελίδας. Το περιγραφόμενο παράδειγμα ορίζει ότι η αρχική εντολή για την σελίδα είναι this.MasterPageFile και μετατρέπεται σε CustomMasterPage.master, με τη χρήση του Page_Prelnit. Αυτή η διαδικασία πρέπει να πραγματοποιηθεί σε αυτή τη φάση, καθώς αργότερα δε θα είναι εφικτό.

Εν κατακλείδι, οι Master Pages προσφέρουν συνέπεια για τις σελίδες ενός ιστοτόπου, μέσω της κοινής διάταξης που μπορούν να παρέχουν στις εξαρτώμενες ιστοσελίδες από ένα συγκεκριμένο master page. Στην ενότητα παρουσιάστηκαν τα χαρακτηριστικά αυτών των σελίδων και ο τρόπος με τον οποίο μπορούν να δημιουργηθούν στο Visual Web Express. Στη συνέχεια περιγράφηκε η σημασία του ρόλου των οδηγιών για τη διαδικασία και το τελικό αποτέλεσμα στο διαδίκτυο, καθώς επίσης και η διάταξη των στοιχείων που περιέχει η εκάστοτε σελίδα. Έπειτα περιγράφηκαν οι Content Pages, τα χαρακτηριστικά και ο τρόπος δημιουργίας τους. Τέλος, αναφέρθηκε η μέθοδος με την οποία υπάρχει η δυνατότητα επέμβασης και αλλαγής των σελίδων, με δυναμικό τρόπο από τον εκάστοτε σχεδιαστή.

4.4 ASP.NET Ajax Controls

Σε αυτήν την ενότητα περιγράφεται η βασική τεχνολογία Ajax της ASP.NET, τα σημαντικότερα χαρακτηριστικά και οι χρήσεις της, χωρίς το σχεδιαστής να γνωρίζει σε βάθος την Javascript και χωρίς να χρειάζεται να γράψει κάποιον κώδικα. Στη συνέχεια θα περιγραφούν οι λειτουργίες των ScriptManager Control, UpdatePanel Control και UpdateProgress Control και ο τρόπος με τον οποίο συνδυάζονται και τα τρία, ώστε να παράγουν μία όμορφη και απλή εμπειρία Ajax όπου ο χρήστης πατώντας ένα πλήκτρο, δεν είναι απαραίτητο να επαναφορτώσει όλη την ιστοσελίδα, μόνο τα επιθυμητά τμήματά της.

Η ενότητα ξεκινάει με τη βασική ορολογία, τα ακρωνύμια και καταλήγει με τις δυνατότητες της AJAX. Αρχικά αναφέρονται οι παραδοσιακές διαδικτυακές εφαρμογές και οι δυνατότητες τους. Η πρώτη λειτουργία είναι η Postback Operation, όπου, όταν ο χρήστης πατάει ένα πλήκτρο, τα δεδομένα επιστρέφουν στον εξυπηρετητή. Τα δεδομένα αυτά υπόκεινται σε επεξεργασία και κατόπιν, επαναφορτώνονται από την αρχική σελίδα ή κάποια άλλη. Έτσι, ολόκληρη η σελίδα ανανεώνεται καθώς υποβάλλονται εκ νέου τα δεδομένα. Σε αντίθεση με αυτή τη λειτουργία, η AJAX δε στοχεύει στην ανανέωση ολόκληρης της σελίδας. Για το λόγο

αυτό ο δημιουργός προβαίνει σε μερική αναβάθμιση σελίδας (Partial-Page Updates), όπου ανανεώνεται ένα τμήμα ή τομέα της σελίδας με το να πατηθεί από τον χρήστη ένα πλήκτρο, ένας υπερσύνδεσμος ή κάποια επιλογή του μενού. Συγκεκριμένα, το Partial-Page Updates είναι σχεδιασμένο ώστε να λαμβάνεται μόνο το τμήμα της πληροφορίας που είναι απαραίτητη στο χρήστη. Στο σημείο εξηγείται ότι το ακρωνύμιο AJAX προκύπτει από το Asynchronous JavaScript and XML και στην πραγματικότητα πρόκειται για μία 'συλλογή πραγμάτων' που λαμβάνουν χώρα παρασκηνιακά ώστε να είναι δυνατή η ανταλλαγή δεδομένων και αναβάθμισης τμήματος των ιστοσελίδων.

Η AJAX είναι ένα ακρωνύμιο το οποίο περιλαμβάνει πολλές τεχνολογίες. Αυτό που λειτουργεί ως συνδετικός κρίκος ανάμεσα σε όλες αυτές είναι το κλειδί όλων των σύγχρονων περιηγητών, το XML. Αυτό το αντικείμενο επιτρέπει τη δημιουργία παράλληλων κλήσεων (asynchronous calls) από μία ιστοσελίδα από και προς τον περιηγητή και τον εξυπηρετητή και τούμπαλιν μέσω του XMLHttpRequest. Το αντικείμενο στέλνει ένα ερώτημα στον εξυπηρετητή και λαμβάνει δεδομένα, χωρίς να είναι απαραίτητη η επαναφόρτωση της σελίδας. Στην πραγματικότητα, απλά φορτώνονται τα δεδομένα XML ή JSON και η διαδικασία ανταλλαγής ελαχιστοποιείται. Με την ASP.NET υπάρχει η δυνατότητα ανταλλαγής δεδομένων μαζί με τα αρχεία XML και οτιδήποτε άλλο, χωρίς περιορισμό. Τέλος, με τον όρο παράλληλη/ασύγχρονη κλήση περιγράφεται μία διαδικασία όπου ο περιηγητής δεν 'κλειδώνεται' όταν γίνεται κλήση προς τον εξυπηρετητή, αντιθέτως λαμβάνει χώρα παρασκηνιακά μία αυτόνομη διαδικασία κλήσης, με αποτέλεσμα ο χρήστης να συνεχίζει να αλληλεπιδρά με άλλες εντολές και διαδικασίες, χωρίς να απομονώνεται σε αναμονή. Τα μηνύματα στέλνονται και επιστρέφονται προς τις δύο κατευθύνσεις χρησιμοποιώντας το XMLHttpRequest, ελαχιστοποιείται ο αριθμός των ανανεώσεων των σελίδων (Partial-Page Updates) και αυτή η διαδικασία μπορεί να ενσωματώνεται σε κάθε διαδικτυακή υπηρεσία ή εφαρμογή REST και να λειτουργεί με κάθε σύγχρονο περιηγητή.

Κάποιες από αυτές τις απαιτούμενες τεχνολογίες περιλαμβάνουν XML, Extensible Markup Language, ο οποίος είναι ένας τρόπος ανταλλαγής δεδομένων και σηματοδότησής τους μέσω ετικετών (tags). Επίσης μπορεί να χρησιμοποιηθεί το JSON, JavaScript Object Notation, το οποίο είναι το σημαντικότερο τμήμα του AJAX και συγκεκριμένα, αποτελεί τον τρόπο με τον οποίο ανταλλάσσονται δεδομένα ανάμεσα σε εξυπηρετητή και φυλλομετρητή με απλότητα και με σηματοδότηση των

δεδομένων. Το DOM, Document Object Model, αντιπροσωπεύει τη μνήμη διάδρασης, όταν χρησιμοποιείται η AJAX. Για παράδειγμα, όταν γίνεται λήψη ενός XML ή JSON μηνύματος, τα δεδομένα αυτά χρησιμοποιούνται στην αναβάθμιση της ιστοσελίδας, γεγονός που προϋποθέτει την ανανέωση του μοντέλου του αντικειμένου του κειμένου. Ένας ακόμα όρος είναι το REST, Representational State Transfer, ο οποίος είναι ένα τρόπος αναγνώρισης πηγών ενός εξυπηρετητή και ανάκτησης των δεδομένων. Το REST είναι κάτι που οι σχεδιαστές δε είναι υποχρεωμένοι να χρησιμοποιούν, αλλά το συναντούν συχνά στον κόσμο της AJAX. Επίσης, τα Web Services είναι παρόμοια του REST.

Σε αυτή το τμήμα περιγράφονται οι βασικές εντολές της ASP.NET AJAX, οι οποίες βελτιώνουν τη λειτουργικότητα του σε μία εφαρμογή. Ο πυρήνας της ASP.NET AJAX είναι η εντολή ScriptManager, η οποία μπορεί να κάνει διάφορα πράγματα, αλλά ένας από τους βασικούς της σκοπούς είναι να φορτώνει τις απαραίτητες εντολοδέσμες που θα επιτρέπουν στην AJAX να χρησιμοποιηθεί σε μία συγκεκριμένη σελίδα. Για το λόγο αυτό, η εντολή του ScriptManager θα εφαρμόζεται πριν από οποιαδήποτε άλλη εντολή όπως οι UpdatePanel, UpdateProgress, ScriptManagerProxy και Timer. Το UpdatePanel χρησιμοποιείται για να πυροδοτήσει ένα αίτημα AJAX, ενώ το ScriptManager για να διαχειριστεί τα διάφορα αιτήματα της AJAX, συμπεριλαμβάνοντας και τους φακέλους της JavaScript. Το UpdateProgress χρησιμοποιείται στην περίπτωση ανάγκης οπτικοποίησης (μέσω ενός γραφήματος για παράδειγμα), όταν ξεκινάει η AJAX, ώστε ο χρήστης που βρίσκεται σε αναμονή να μην αναρωτιέται για τις δραστηριότητες που λαμβάνουν χώρα παρασκηνιακά. Η εντολή Timer μπορεί να χρησιμοποιηθεί για να πυροδοτήσει τις AJAX calls ανά συγκεκριμένα χρονικά διαστήματα και τέλος, το ScriptManagerProxy είναι χρήσιμο στην περίπτωση που πρέπει να τοποθετηθούν περισσότερες, από τη βασική ScriptManager, εντολοδέσμες σε σελίδες, οι οποίες ακολουθούν τη διάταξη μίας Masterpage.

Το ScriptManagerControl είναι το πρώτο στοιχείο που πρέπει να προστεθεί σε μία σελίδα, στην περίπτωση εργασίας με AJAX, καθώς αποτελεί το βασική εντολή του περιβάλλοντος AJAX, η οποία λειτουργεί ως διαχειριστής και συντονιστής όλων των εντολοδέσμων που χρησιμοποιούνται από μία σελίδα. Είναι επίσης επιφορτισμένο με το να φορτώνει τις εντολοδέσμες που απαιτούνται τις AJAX calls καθώς επίσης και συνηθισμένες. Τέλος, έχει την ικανότητα να δημιουργεί αντικείμενα από την πλευρά των χρηστών και να συντονίζει ασύγχρονες επιχειρήσεις. Υπάρχει η δυνατότητα

εισαγωγής του ScriptManager είτε ΄σέρνοντάς’ το μες τη σελίδα από τη γραμμή εργαλείων, είτε χειροκίνητα με σύνταξη του αντίστοιχου κώδικα στην πηγή του (<asp:ScriptManager id=’’sm’’ runat+’’server’’ />). Ένα πολύ σημαντικό ζήτημα που αφορά στην απομάκρυνση ιών είναι ότι οι αναβαθμίσεις των partial-pages μπορούν να απενεργοποιηθούν μέσω του ScriptManager;s EnablePartialRendering property.

Το επόμενο ζήτημα μελέτης είναι η εντολή UpdatePanel η οποία με ελάχιστη ανάγκη κωδικοποίησης μπορεί να ενεργοποιήσει τις AJAZ calls, χωρίς ο σχεδιαστής να είναι ιδιαίτερος γνώστης του JavaScript. Ορισμένα από τα χαρακτηριστικά του είναι η ικανότητα να ενεργοποιεί τμήματα μίας σελίδας, να διαχειρίζεται δράσεις που πυροδοτούνται από στοιχείο XTML, να υποστηρίζει ρυθμίσεις UpdatePanel και να παρέχει μία ταμπλέτα που ονομάζεται Content Template και αποτελεί την εξωτερική επιφάνεια της ταμπλέτας που εφαρμόζεται σε μία σελίδα. Η εντολή αυτή λειτουργεί με το να μετατρέπει απλές παρασκηνιακές λειτουργίες σε ασύγχρονες, μέσω του XMLHttpRequest. Επίσης ελέγχει την ταμπλέτα από τη στιγμή που τα ViewState επιστρέφονται στον εξυπηρετητή, απαντά με μήνυμα δεδομένων απευθείας στη σελίδα και δεν είναι απαραίτητη η γνώση JavaScript για τη λειτουργία του

Τέλος, παρουσιάζεται η εντολή UpdateProgress η οποία είναι πολύ απλή στη χρήση και εξαιρετικά παραγωγική στο να ενημερώνει το χρήστη για τη στιγμή που ξεκινάει και τελειώνει η AJAX call. Κάποια από τα βασικά χαρακτηριστικά της είναι ότι παρέχει στους χρήστες ιστορικό μετάδοσης των αιτημάτων τους και επίσης, χρησιμοποιεί μία ProgressTemplate για να ορίσει την πρόοδο του αιτήματος.

Συμπερασματικά, οι αναβαθμίσεις των Partial-page επιτρέπουν σε τμήματα σελίδων να φορτώσουν χωρίς να ανανεώσουν το σύνολο της σελίδας. Επίσης, η ASP.NET παρέχει ένα σύνολο από εντολές που ενεργοποιούνται μέσω της AJAX και απλοποιούν τη διαδικασία των Partial-page. Βασικά εργαλεία είναι τα ScriptManager, UpdatePanel και UpdateProgress.

4.5 Δουλεύοντας με τα Data Source Controls

Σε αυτήν την ενότητα παρουσιάζεται ο τρόπος με τον οποίο μπορούν να χρησιμοποιηθούν οι ρυθμίσεις στις πηγές δεδομένων (Data Source Controls) στις ASP.NET Web Forms Applications. Το κείμενο ξεκινά με τρόπους σύνδεσης δεδομένων (data binding), το λόγο που τη χρησιμοποιούν οι σχεδιαστές και συγκεκριμένα, το γεγονός ότι πρόκειται για ένα εξαιρετικά παραγωγικό εργαλείο το

οποίο επιτρέπει τη σύνδεση δεδομένων με περιορισμένη χρήση γραφής ενός κώδικα και συνεχίζει με ορισμένες διατυπώσεις σύνδεσης δεδομένων (data binding expressions). Εν τέλει, παρουσιάζονται ορισμένα εργαλεία όπως το SqlDataSource, το ObjectDataSource, το EntityDataSource και το QueryExtender.

Το Data Binding επιτρέπει την εύρεση και χρήση δεδομένων από διάφορες πηγές δεδομένων, αντικείμενα, πλατφόρμες κτλ.. Έχει δημιουργηθεί στο περιβάλλον της ASP.NET και επιτρέπει τη σύνδεση δεδομένων χωρίς τη δημιουργία 'θηλιών' ανάμεσα τους, όπως συνέβαινε παλαιότερα. Για το λόγο αυτό, έχει ελαχιστοποιηθεί η ανάγκη για δημιουργία εκτεταμένου και σύνθετου κώδικα. Σε αυτή τη διαδικασία βοηθάει ο Data Source Control ο οποίος διαχειρίζεται τα δεδομένα, επιτρέπει την ελεγχόμενη είσοδό τους από διάφορες πηγές και δημιουργεί δεσμούς στις ρυθμίσεις, ακολουθώντας διάφορους τύπους όπως συγκριτικά, ιεραρχικά ή τυχηματικά δεδομένα. Ένα άλλο στοιχείο είναι οι ρυθμίσεις ενημέρωσης δεδομένων (Data aware controls) οι οποίες έχουν μία ειδική ταυτότητα (DataSourceID) που επιτρέπει την ένωση μίας εντολής με τον έλεγχο δεδομένων. Υπάρχουν αρκετοί τύποι Πηγών δεδομένων όπως η βάση δεδομένων της Access, της SQL, το περιβάλλον Entity το οποίο χρησιμοποιείται για τη δημιουργία μοντέλων Entity, LINQ, Object το οποίο χρησιμοποιείται για δημιουργία 'αρχιτεκτονικών' μοντέλων με αρκετά επίπεδα, το Site Map για πλοήγηση και XML Files.

Τα Data Binding Expressions είναι ένα σημαντικό τμήμα της σύνδεσης αρχείων στην ASP.NET, το οποίο επιτρέπει τη σύνδεση δεδομένων και ρυθμίσεων με τρόπο απλό, γρήγορο και παραγωγικό, καθώς δημιουργεί 'νήματα' σύνδεσης ανάμεσα σε πολλές σελίδες και πηγές εντοπισμού δεδομένων. Ορισμένες από τις εκφράσεις σύνδεσης είναι: <%# Eval (…)%>, <%# Bind (…)%>, <%# XPath (…)%> (π.χ. <%# Eval (''FieldName)%>.

Το επόμενο στάδιο αναφέρεται στο SqlDataSource το οποίο μπορεί να αναζητά δεδομένα σε πολλές διαφορετικές πηγές και να δημιουργεί ένα σύνδεσμο ανάμεσα στις πηγές και τις ρυθμίσεις στην ιστοσελίδα. Υποστηρίζει τέσσερις παρόχους (SQL, OleDB, ODBC, Oracle), έχει τη δυνατότητα να ρέει τα δεδομένα ή να τα φορτώνει στη μνήμη με το να αλλάζει τις ιδιότητές τους, υποστηρίζει το φιλτράρισμα και την ταξινόμηση των δεδομένων, την επιλογή, εισαγωγή, αναβάθμιση και διαγραφή των εντολών και προσφέρει την τρέχουσα υποστήριξη για την ανανέωση της μνήμης. Οι διαφορετικές πηγές δεδομένων υποστηρίζουν διαφορετικές παραμέτρους και ορισμένες από αυτές είναι:

Parameter: αντιπροσωπεύει μία παράμετρο σε ένα παραμετροποιημένο ερώτημα SQL, μία φιλτραρισμένη έκφραση ή ένα αντικείμενο.

CookieParameter: συνδέει την αξία ενός cookie που προέρχεται από τον χρήστη με τον έλεγχο της πηγής δεδομένων.

FormParameter: Συνδέει την αξία μίας φόρμας στη Form Collection με τον έλεγχο της πηγής δεδομένων.

ProfileParameter: Συνδέει την αξία ενός Profile στην ASP.NET με τον έλεγχο της πηγής δεδομένων.

QueryStringParameter: Συνδέει την αξία ενός QueryString με τον έλεγχο της πηγής δεδομένων.

SessionParameter: Συνδέει την αξία ενός Session με τον έλεγχο της πηγής δεδομένων.

Συγκεκριμένα, οι πληροφορίες μπορούν να αντληθούν από παραμέτρους ρυθμίσεων ή απο cookies ή από την επαναληψιμότητα επίσκεψης μίας ιστοσελίδα είτε από διάφορα άλλα στοιχεία.

Παρόλο που το SqlDataSource προσφέρει έναν πολύ παραγωγικό τρόπο για την εισαγωγή και εξαγωγή δεδομένων στην εφαρμογή, δεν εμφανίζει μεγάλη συχνότητα χρήσης και γραφής κώδικα και υπάρχει ο κίνδυνος της διπλοεφαρμογής ενός κώδικα σε περισσότερες περιπτώσεις. Μία εναλλακτική, στην περίπτωση που ο δημιουργός δε χρειάζεται πληροφορίες άμεσα από τη βάση δεδομένων, είναι να χρησιμοποιήσει το ObjectDataSource. Αυτή η εναλλακτική, αντί να συνδέει άμεσα δεδομένα με τις πηγές, δημιουργεί ένα ‘αρχιτεκτονικό’ περιβάλλον με διάφορες στρώσεις και επίπεδα άντλησης δεδομένων, γεγονός που δημιουργεί αμεσότητα σε περίπτωση ανάγκης αλλαγών. Με το ObjectDataSource οι αλλαγές μπορούν να πραγματοποιηθούν άμεσα σε όλα τα επίπεδα εφαρμογής, σε όλες τις ιστοσελίδες σε αντιδιαστολή με το προηγούμενο μοντέλο, όπου όλα γίνονται χειροκίνητα από το σχεδιαστή. Επίσης υποστηρίζει σύνδεση με συνηθισμένα αντικείμενα, όπως επιχειρηματικά, επιτρέπει σε N-Tier/N-Layer να διαχωρίζει τις στρώσεις της παρουσίασης, πληροφόρησης και επιχείρησης, υποστηρίζει επίσης το φιλτράρισμα και την ταξινόμηση των δεδομένων, την επιλογή, εισαγωγή, αναβάθμιση και διαγραφή των εντολών και προσφέρει την τρέχουσα υποστήριξη για την ανανέωση της μνήμης, όπως και η προηγούμενη εφαρμογή. Το εργαλείο ObjectDataSource επιτρέπει σε αντικείμενα δεδομένων να δεθούν με την ASP.NET και στο συγκεκριμένο παράδειγμα το ObjectDataSource ενώνεται σε επίπεδο δεδομένων με το Gridview με την προσθήκη της εντολής

SelectMethod που αντιπροσωπεύει τη μέθοδο με την οποία επιλέγει ο δημιουργός και με Typename=DAL, το οποίο αντιπροσωπεύει την κατηγορία δεδομένων που ενδιαφέρει το δημιουργό. Εντός του DAL, βρίσκονται πολλές διαφορετικές κατηγορίες δεδομένων, ο συνδυασμός των οποίων μοντελοποιείται και ελέγχεται από το ObjectDATASOURCE.

Επόμενο εργαλείο είναι το EntityDataSource το οποίο επιτρέπει την παραγωγή ερωτημάτων LINQ ενάντια σε μία πηγή δεδομένων και συνεργάζεται με το περιβάλλον Entity. Επίσης, υποστηρίζει το φιλτράρισμα και την ταξινόμηση των δεδομένων, την επιλογή, εισαγωγή, αναβάθμιση και διαγραφή των εντολών και προσφέρει την τρέχουσα υποστήριξη για την ανανέωση της μνήμης, όπως και οι προηγούμενες εφαρμογές. Τα βασικά βήματα για να χρησιμοποιήσει ο χρήστης το LinqDataSource είναι να δημιουργήσει ένα μοντέλο του Περιβάλλοντος Entity στο Visual Studio. Έπειτα, πρέπει να ρυθμιστεί το EntityDataSource στη σελίδα του σχεδιαστή για να αναγνωρίσει το ερώτημα, τουτέστιν έναν πίνακα και τέλος, να προσθέσει τις παραμέτρους και/ή τα απαιτούμενα φίλτρα.

Το τελευταίο ζήτημα που θα περιγραφεί είναι το QueryExtender το οποίο δημιουργεί φίλτρα για δεδομένα που έχουν ανακτηθεί από κάποια πηγή δεδομένων. Χρησιμοποιείται σε συνδυασμό με το LinqDataSource ή με το EntityDataSource και εξαιτίας του γεγονότος ότι λειτουργεί με φίλτρα, δεν είναι απαραίτητο για το πρόγραμμα να γνωρίζει τις πηγές δεδομένων και επιτρέπει να είναι εμφανείς διαφορετικές απόψεις των δεδομένων.

Εν κατακλείδι, αυτό το τμήμα του κειμένου επικεντρώθηκε στον τρόπο με τον οποίο εισάγουμε δεδομένα στις εφαρμογές ASP.NET και τις ρυθμίσεις που απαιτούνται για τις συνδέσεις των δεδομένων, όπου η βασική σύνταξη σύνδεσης περιλαμβάνει <%#Eval(..)%> και μία μορφή των νημάτων σύνδεσης μπορεί να είναι <%$ConnectionStrings:YourConnSting%>. Επίσης αναφέρθηκαν οι βασικές ρυθμίσεις του ελέγχου των πηγών δεδομένων, όπως τα SqlDataSource και EntityDataSource και ο τρόπος με τον οποίον ελαχιστοποιούν την δουλειά των δημιουργών από τη στιγμή που έχουν εισαχθεί τα δεδομένα. Τέλος, έγινε κατανοητό ότι αρκετές εφαρμογές, όπως το Gridview μπορούν να χρησιμοποιηθούν για να παράγουν HTML για πολλαπλές ροές δεδομένων και ότι τα ερωτήματα Entity μπορούν να επεκταθούν με τη χρήση του QueryExtender.

4.6 ASP.NET Web Forms και προκλήσεις

Ο προγραμματισμός web εφαρμογών παρουσιάζει προκλήσεις. Μεταξύ των προκλήσεων είναι:

- Υλοποίηση μιας πλούσιας web διεπαφής χρήστη - Μπορεί να είναι δύσκολο και κουραστικό να σχεδιαστεί και να εφαρμοστεί ένα περιβάλλον εργασίας χρήστη, χρησιμοποιώντας βασικές εγκαταστάσεις HTML, ειδικά εάν η σελίδα έχει σύνθετη διάταξη, μεγάλη ποσότητα δυναμικού περιεχομένου, και πολλά διαδραστικά αντικείμενα με πλήρεις δυνατότητες.
- Διαχωρισμός πελάτη και διακομιστή- Σε μια εφαρμογή Web, ο πελάτης (browser) και ο server είναι διαφορετικά προγράμματα που συχνά εκτελούνται σε διαφορετικούς υπολογιστές (ακόμα και σε διαφορετικά λειτουργικά συστήματα). Κατά συνέπεια, τα δύο μισά της εφαρμογής μοιράζονται πολύ λίγες πληροφορίες, μπορούν να επικοινωνούν, αλλά συνήθως ανταλλάσσουν μόνο μικρά κομμάτια απλών πληροφοριών.
- stateless εκτέλεση - Όταν ένας διακομιστής web λαμβάνει μια αίτηση για μια σελίδα, βρίσκει τη σελίδα, την επεξεργάζεται, τη στέλνει στο πρόγραμμα περιήγησης, και στη συνέχεια απορρίπτει όλες τις πληροφορίες της σελίδας. Εάν ο χρήστης ζητά ξανά την ίδια σελίδα, ο διακομιστής επαναλαμβάνει ολόκληρη την ακολουθία, επεξεργάζοντας ξανά τη σελίδα από το μηδέν. Με άλλα λόγια, ένας διακομιστής δεν έχει καμία μνήμη των σελίδων που έχει επεξεργαστεί, δηλαδή οι σελίδες είναι stateless. Ως εκ τούτου, εάν μια εφαρμογή πρέπει να διατηρήσει τις πληροφορίες για μια σελίδα, η stateless φύση της μπορεί να αποτελέσει πρόβλημα.
- Άγνωστες δυνατότητες πελάτη - Σε αρκετές περιπτώσεις, οι web εφαρμογές είναι προσιτές σε πολλούς χρήστες που χρησιμοποιούν διαφορετικά προγράμματα περιήγησης. Τα προγράμματα περιήγησης έχουν διαφορετικές ικανότητες, γεγονός που καθιστά δύσκολο τη δημιουργία μιας εφαρμογής που θα τρέχει εξίσου καλά σε όλα αυτά.
- Επιπλοκές με την πρόσβαση στα δεδομένα – Η ανάγνωση και η εγγραφή σε μια πηγή δεδομένων σε παραδοσιακές web εφαρμογές μπορεί να είναι περίπλοκη και απαιτητική από άποψη πόρων.

Η αντιμετώπιση αυτών των προκλήσεων για web εφαρμογές μπορεί να απαιτήσει σημαντικό χρόνο και προσπάθεια. Οι ASP.NET Web Forms και το πλαίσιο ASP.NET απευθύνονται στις προκλήσεις αυτές με τους εξής τρόπους:

- Συνεπές μοντέλο αντικειμένων – Το ASP.NET πλαίσιο σελίδων παρέχει ένα μοντέλο αντικειμένων που επιτρέπει στον προγραμματιστή να σκεφτεί τις φόρμες του ως μονάδα και όχι ως ξεχωριστά κομμάτια πελάτη και διακομιστή. Σε αυτό το μοντέλο, μπορεί να προγραμματίσει τη σελίδα περισσότερο προετοιμασμένος από ότι στις παραδοσιακές web εφαρμογές, συμπεριλαμβανομένης της δυνατότητας να ορίσει τις ιδιότητες για τα στοιχεία της σελίδας και να αποκριθεί σε γεγονότα. Επιπλέον, μπορούν να χρησιμοποιηθούν server controls με τον τρόπο που θα μπορούσαν να λειτουργήσουν σε μια εφαρμογή πελάτη, χωρίς να χρειάζεται η ενασχόληση με τη δημιουργία HTML για να παρουσιάσει και να επεξεργαστεί τα στοιχεία ελέγχου και το περιεχόμενό τους.
- Event-driven μοντέλο προγραμματισμού- Οι ASP.NET Web Forms φέρουν στις web εφαρμογές το γνωστό μοντέλο της συγγραφής χειριστών γεγονότων (event handlers) για συμβάντα που λαμβάνουν χώρα είτε στον πελάτη είτε στον διακομιστή. Το πλαίσιο ASP.NET αφαιρεί αυτό το μοντέλο με τέτοιο τρόπο ώστε ο υποκείμενος μηχανισμός της σύλληψης ένα γεγονότος για τον πελάτη, η μετάδοση στον server, και η κλήση της κατάλληλης μεθόδου είναι όλα αυτόματα και αόρατα στον προγραμματιστή. Το αποτέλεσμα είναι μια διαυγής, εύκολα γραμμένη δομή κώδικα που υποστηρίζει το συμβάν με γνώμονα την ανάπτυξη.
- Αυτόματη Διαχείριση Κατάστασης – Το πλαίσιο σελίδων ASP.NET χειρίζεται αυτόματα το έργο της διατήρησης της κατάστασης της σελίδας και των ελεγκτών της, και προσφέρει σαφής τρόπους για τη διατήρηση της κατάστασης των πληροφοριών για συγκεκριμένες εφαρμογές. Αυτό επιτυγχάνεται χωρίς τη βαριά χρήση των πόρων του server και μπορεί να υλοποιηθεί με ή χωρίς την αποστολή cookies στον browser.
- Υποστήριξη του .NET Framework common language runtime (CLR) – Το πλαίσιο σελίδων ASP.NET είναι χτισμένο στο πλαίσιο .NET, έτσι ώστε το σύνολο του πλαισίου να είναι διαθέσιμο σε οποιαδήποτε εφαρμογή ASP.NET. Οι εφαρμογές μπορεί να γραφτούν σε οποιαδήποτε γλώσσα που είναι

συμβατή με το χρόνο εκτέλεσης. Επιπλέον, η πρόσβαση στα δεδομένα έχει απλοποιηθεί με τη χρήση της υποδομής πρόσβασης στα δεδομένα που παρέχεται από το πλαίσιο .NET, συμπεριλαμβανομένου του ADO.NET.

- Κλιμάκωση απόδοσης server – Το πλαίσιο σελίδων ASP.NET δίνει τη δυνατότητα κλιμάκωσης μιας web εφαρμογής από έναν υπολογιστή με ένα μόνο επεξεργαστή σε ένα πολύ-υπολογιστικό σύστημα web καθαρά και χωρίς πολύπλοκες αλλαγές στην λογική της εφαρμογής.

4.7 Web Forms ή MVC

Ο προγραμματιστής θα πρέπει να εξετάσει προσεκτικά εάν θα υλοποιήσει μια εφαρμογή web χρησιμοποιώντας είτε τις ASP.NET Web Forms ή ένα άλλο μοντέλο, όπως το πλαίσιο ASP.NET MVC. Το πλαίσιο MVC δεν αντικαθιστά μοντέλο Web Forms. Μπορεί να χρησιμοποιηθεί οποιοδήποτε πλαίσιο από τα δύο για web εφαρμογές. Πριν αποφασίσει να χρησιμοποιήσει το μοντέλο Web Forms ή το πλαίσιο MVC για μια συγκεκριμένη Web εφαρμογή, θα πρέπει να ζυγίσει τα πλεονεκτήματα της κάθε προσέγγισης.

Παρακάτω, θα προσπαθήσουμε να παραθέσουμε τα πλεονεκτήματα μιας Web εφαρμογής που χρησιμοποιεί Web Forms και μιας Web εφαρμογής που χρησιμοποιεί το μοντέλο MVC.

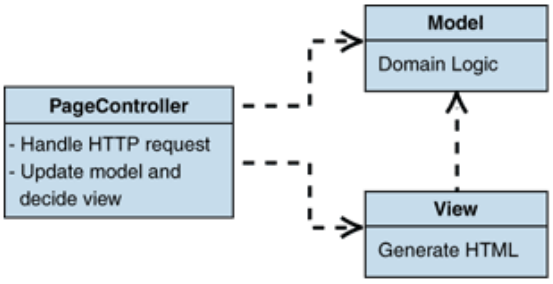
4.7.1 Πλεονεκτήματα μιας Web εφαρμογής βασισμένης σε Web Forms

Όσον αφορά μια Web εφαρμογή που βασίζεται σε Web Forms , τα πλεονεκτήματα είναι:

- Υποστηρίζει ένα μοντέλο γεγονότων που διατηρεί την κατάσταση μέσω HTTP, η οποία ωφελεί την ανάπτυξη εφαρμογών Web. Η εφαρμογή που βασίζεται σε Web Forms παρέχει δεκάδες γεγονότα (events) που υποστηρίζονται σε εκατοντάδες ελεγκτές (server controls).
- Χρησιμοποιεί έναν *Page Controller* που προσθέτει λειτουργικότητα σε μεμονωμένες σελίδες[4]. Ο Page Controller (Εικόνα 9) δέχεται είσοδο από το αίτημα σελίδας, επικαλείται τις αιτούμενες δράσεις στο μοντέλο και καθορίζει ποια είναι η σωστή προβολή για να εμφανιστεί η σελίδα.

[4] https://msdn.microsoft.com/en-us/library/ms978764.aspx

- Λειτουργεί καλά για μικρές ομάδες web developers και σχεδιαστών που θέλουν να επωφεληθούν από το μεγάλο αριθμό των προϊόντων που διατίθενται για την ταχεία ανάπτυξη εφαρμογών.
- Γενικώς, είναι λιγότερο πολύπλοκο για την ανάπτυξη εφαρμογών, επειδή τα συστατικά (Page class, controls, κτλ) είναι ολοκληρωμένα και συνήθως απαιτούν λιγότερο κώδικα από το μοντέλο MVC.

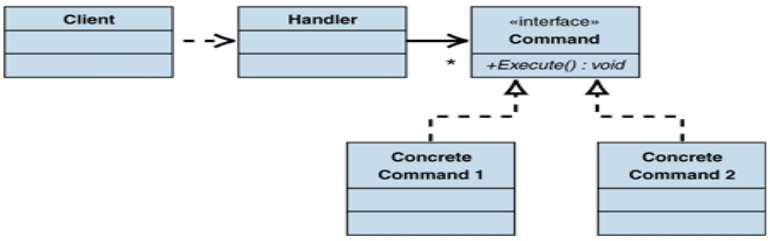


Εικόνα 9: Η δομή του ελεγκτή σελίδας (page controller).

4.7.2 Πλεονεκτήματα μίας Web εφαρμογής βασισμένης σε MVC

Το πλαίσιο ASP.NET MVC προσφέρει τα ακόλουθα πλεονεκτήματα:

- Διευκολύνει τη διαχείριση της πολυπλοκότητας διαιρώντας μια εφαρμογή στο μοντέλο, την προβολή και τον ελεγκτή.
- Δεν χρησιμοποιεί την κατάσταση της προβολής ή φόρμες που βασίζονται σε server. Αυτό κάνει το MVC πλαίσιο ιδανικό για τους προγραμματιστές που θέλουν πλήρη έλεγχο της συμπεριφοράς της εφαρμογής.
- Χρησιμοποιεί έναν Front Controller που επεξεργάζεται τα αιτήματα της Web εφαρμογής. Ο Front Controller δημιουργεί ένα κανάλι μετάδοσης όλων των αιτημάτων και συνήθως υλοποιείται σε δύο μέρη: τον χειριστή και την ιεραρχία των εντολών (**Error! Reference source not found.**0).
- Παρέχει καλύτερη υποστήριξη για ανάπτυξη TDD (Test Driven Development).
- Λειτουργεί καλά για web εφαρμογές που υποστηρίζονται από μεγάλες ομάδες προγραμματιστών και web σχεδιαστών και χρειάζονται υψηλό βαθμό ελέγχου επί της συμπεριφοράς της εφαρμογής.



Εικόνα 107: Η δομή του Front Controller.

4.8 Δημιουργία μιας απλής σελίδας με το ASP.NET Web Forms στο Visual Studio 2013

Σε αυτήν την ενότητα θα γίνει μια εισαγωγή στο περιβάλλον ανάπτυξης ιστοσελίδων του Microsoft Visual Studio 20135 και του Microsoft Visual Studio Express 2013 for Web 6 (τα οποία στη συνέχεια θα αναφέρονται απλά ως Visual Studio). Θα παρουσιαστεί ο τρόπος δημιουργίας μιας απλής σελίδας ASP.NET Web Forms και θα επιδειχθούν οι βασικές τεχνικές της δημιουργίας μιας νέας σελίδας, προσθέτοντας controllers και κώδικα.

Οι λειτουργίες που θα παρουσιαστούν είναι:

- Δημιουργία ενός project Web Forms εφαρμογής
- Εργασία με το Visual Studio.
- Δημιουργία μιας ASP.NET σελίδας.
- Πρόσθεση controllers.
- Πρόσθεση event handlers.
- Εκτέλεση και έλεγχος της σελίδας από το Visual Studio.

Για την παρουσίαση που ακολουθεί, θεωρείται ότι έχει επιλεχθεί η συλλογή ρυθμίσεων Web Development του Visual Studio7.

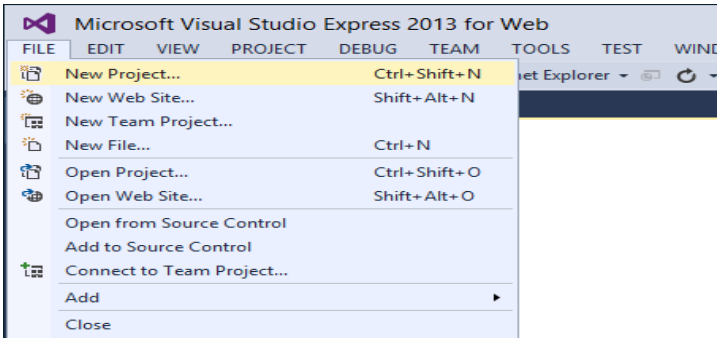
4.8.1 Δημιουργία ενός έργου web εφαρμογής και μιας σελίδας

Σε αυτό το κομμάτι, θα δημιουργηθεί ένα έργο web εφαρμογής και θα προστεθεί μια νέα σελίδα σε αυτό. Θα προστεθεί επίσης κείμενο HTML και θα εκτελεστεί η σελίδα στον browser.

[5] https://www.visualstudio.com/downloads/download-visual-studio-vs#vs
[6] https://www.visualstudio.com/downloads/download-visual-studio-vs#express-web
[7] https://msdn.microsoft.com/library/ff521558.aspx

4.8.2 Δημιουργία έργου web εφαρμογής

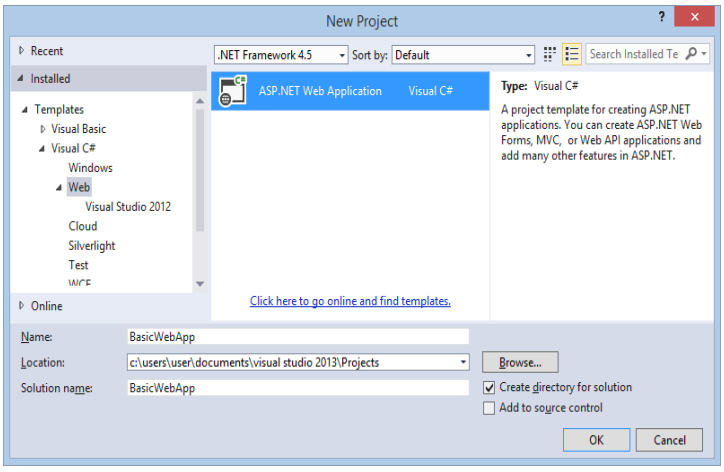
- Άνοιγμα του Microsoft Visual Studio.
- Στο μενού *File*, επιλογή *New Project…* (**Error! Reference source not found.**1)



Εικόνα 8: Δημιουργία νέου project.

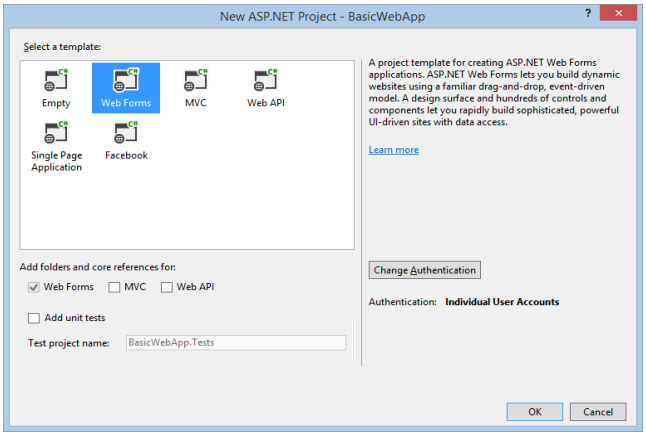
Το παράθυρο διαλόγου New Project εμφανίζεται.

- Επιλογή ομάδας προτύπων Templates -> Visual C# -> Web στα αριστερά.
- Επιλογή του ASP.NET Web Application template στο κέντρο της στήλης.
- Ονομασία του project BasicWebApp και OK (**Error! Reference source not found.**2).



Εικόνα 9: Παράθυρο διαλόγου νέου project.

Στη συνέχεια επιλέγουμε το πρότυπο Web Forms και με το πάτημα του κουμπιού OK, δημιουργείται το project (Εικόνα 13).



Εικόνα 10: Επιλογή προτύπου Web Forms.

Το Visual Studio δημιουργεί ένα νέο έργο που περιλαμβάνει την προδιαγεγραμμένη λειτουργικότητα που παρέχει το μοντέλο web forms. Όχι μόνο παρέχει μια σελίδα Home.aspx, μια σελίδα About.aspx και μια σελίδα Contact.aspx, αλλά περιλαμβάνει επίσης τη λειτουργία μελών, η οποία καταγράφει τους χρήστες και αποθηκεύει τα διαπιστευτήριά τους, ώστε να μπορούν να συνδεθούν με την ιστοσελίδα. Όταν μια νέα σελίδα δημιουργείται, από προεπιλογή, το Visual Studio εμφανίζει τη σελίδα σε source view, όπου μπορεί κανείς να δει τα HTML στοιχεία της σελίδας.

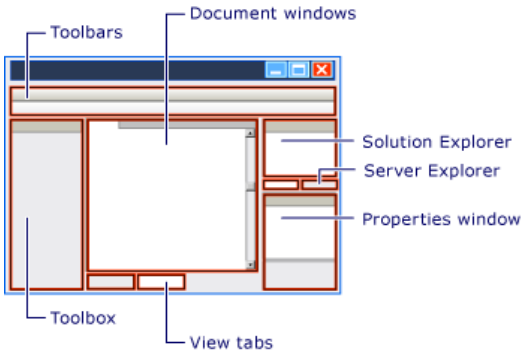
4.8.3 Το Περιβάλλον Ανάπτυξης του Visual Studio

Στην ***Error! Reference source not found.***4 φαίνονται τα συχνότερα χρησιμοποιούμενα παράθυρα και εργαλεία. Αυτά είναι τα εξής:

- Γραμμές εργαλείων (Toolbars). Παρέχουν εντολές για τη μορφοποίηση κειμένου, εύρεση κειμένου, και ούτω καθεξής. Ορισμένες γραμμές εργαλείων είναι διαθέσιμες μόνο σε προβολή σχεδίασης.
- Παράθυρο Solution Explorer. Εμφανίζει τα αρχεία και τους φακέλους στην web εφαρμογή.
- Παράθυρο εγγράφων (Document Window**).** Εμφανίζει τα έγγραφα εργασίας σε καρτέλες παραθύρων. Η εναλλαγή μεταξύ εγγράφων γίνεται με κλικ στις καρτέλες.
- Παράθυρο ιδιοτήτων (Properties Window). Επιτρέπει την αλλαγή των ρυθμίσεων για τη σελίδα, τα στοιχεία HTML, τους ελέγχους, και άλλα αντικείμενα.
- Καρτέλες Προβολής (View Tabs). Παρουσιάζουν διαφορετικές όψεις του ίδιου εγγράφου. Η προβολή *Σχεδίασης* (Design) αποτελεί μια επιφάνεια

επεξεργασίας What You See Is What You Get (WYSIWYG). Η προβολή Source αποτελεί τον επεξεργαστή HTML της σελίδας. Η προβολή Split απεικονίζει και την προβολή Σχεδίασης και την προβολή Source για το έγγραφο.

- Εργαλειοθήκη (ToolBox). Παρέχει ελέγχους και στοιχεία HTML που μπορούν να συρθούν στην σελίδα. Τα στοιχεία της εργαλειοθήκη ομαδοποιούνται ανά λειτουργία.
- Server Explorer. Εμφανίζει συνδέσεις βάσης δεδομένων. Αν ο Server Explorer δεν είναι ορατός, από το μενού προβολής εμφανίζεται με την επιλογή server explorer.



Εικόνα 11: Το περιβάλλον του Visual Studio.

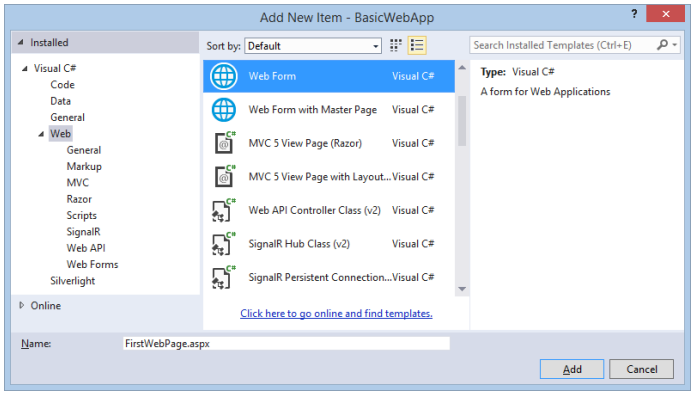
4.8.4 Δημιουργία μιας νέας σελίδας ASP.NET Web Forms

Όταν δημιουργείται μια νέα Web Form εφαρμογή χρησιμοποιώντας το πρότυπο ASP.NET Web Application, το Visual Studio προσθέτει μια σελίδα ASP.NET (Web Forms page) που ονομάζεται Default.aspx, καθώς και διάφορα άλλα αρχεία και φακέλους. Η σελίδα Default.aspx μπορεί να χρησιμοποιηθεί ως αρχική σελίδα για την web εφαρμογή. Ωστόσο, σε αυτή την ενότητα, θα δημιουργηθεί μια νέα σελίδα.

Για την προσθήκη μιας σελίδας στην web εφαρμογή:

- Πρέπει καταρχήν να κλείσει η σελίδα Default.aspx. Αυτό μπορεί να γίνει με κλικ στην καρτέλα που εμφανίζει το αρχείο με το όνομα και κλικ στην επιλογή κλεισίματος.
- Στον Solution Explorer, δεξί-κλικ στο όνομα της web εφαρμογής (στο παράδειγμα που παρουσιάζεται το όνομα της εφαρμογής είναι BasicWebSite), και έπειτα Add → New Item. Το παράθυρο διάλογου Add New Item εμφανίζεται.

- Επιλογή της ομάδας προτύπων Visual C# → Web στα αριστερά. Στην συνέχεια, επιλογή Web Form από την μεσαία λίστα και ονομασία της σελίδας ως FirstWebPage.aspx (**Error! Reference source not found.**5).
- Με κλικ στο Add η ιστοσελίδα προστίθεται στο έργο. Το Visual Studio δημιουργεί την καινούργια σελίδα και την ανοίγει.

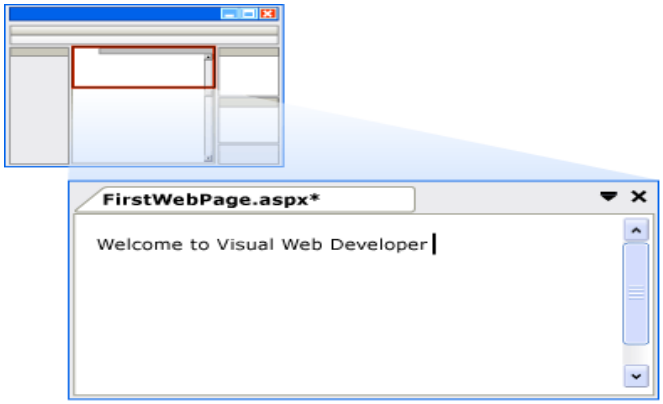


Εικόνα 12: *Δημιουργία Web Form σελίδας.*

4.8.5 Προσθήκη HTML στην σελίδα

Σε αυτή την ενότητα θα προστεθεί κάποιο στατικό κείμενο στη σελίδα. Για να προστεθεί κείμενο στη σελίδα:

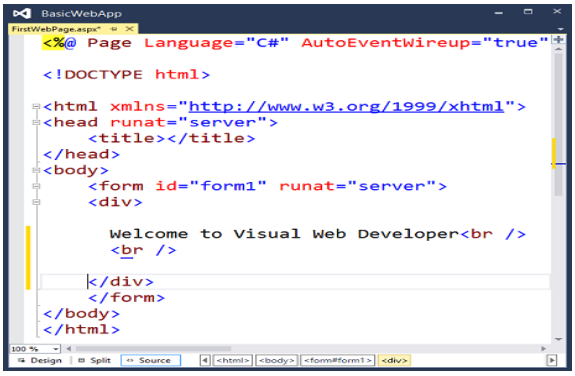
- Μέσω της καρτέλας Design στο κάτω μέρος του παραθύρου, μπορεί να γίνει εναλλαγή στην προβολή σχεδίασης. Η προβολή σχεδίασης απεικονίζει την τρέχουσα σελίδα με έναν WYSIWYG τρόπο. Σε αυτό το σημείο δεν υπάρχει ούτε κείμενο ούτε κάποιο στοιχείο ελέγχου στη σελίδα, οπότε η σελίδα είναι κενή, με εξαίρεση μια γραμμή που σχηματίζει ένα ορθογώνιο. Το ορθογώνιο αναπαριστά ένα στοιχείο div στη σελίδα.
- Κλικ στο ορθογώνιο που φαίνεται σκιαγραφημένο, πληκτρολόγηση της φράσης Welcome to Visual Web Developer και πάτημα Enter δύο φορές.



Εικόνα 13: Κείμενο σε προβολή σχεδίασης (design view).

Στην Εικόνα 16 φαίνεται το κείμενο που πληκτρολογήθηκε σε προβολή σχεδίασης.

Με μετάβαση στην προβολή Source, φαίνεται ο πηγαίος κώδικας HTML που δημιουργείται κατά την πληκτρολόγηση σε προβολή σχεδίασης (**Error! Reference source not found.**7).



Εικόνα 14: Κώδικας HTML σε προβολή Source.

4.8.6 Εκτέλεση της σελίδας

Πριν προστεθούν στοιχεία ελέγχου στη σελίδα, μπορεί πρώτα να εκτελεστεί. Για να τρέξει η σελίδα:

- Στον Solution Explorer, δεξί-κλικ στο FirstWebPage.aspx και επιλογή Set as Start Page.
- Με CTRL+F5 η σελίδα εκτελείται.

Η σελίδα εμφανίζεται στον browser. Παρά το γεγονός ότι η σελίδα που δημιουργήθηκε έχει επέκταση ονόματος .aspx, τρέχει προς το παρόν, όπως οποιαδήποτε σελίδα HTML. Για να εμφανιστεί μια σελίδα στο πρόγραμμα περιήγησης, μπορεί επίσης να γίνει δεξί κλικ στη σελίδα στον Solution Explorer και να επιλεγεί η προβολή σε browser. Με κλείσιμο του browser, η εφαρμογή σταματάει.

4.9 Προσθήκη και προγραμματισμός στοιχείων ελέγχου

Ο προγραμματιστής μπορεί να προσθέσει στη σελίδα server controls. Τα server controls, όπως κουμπιά, ετικέτες, πλαίσια κειμένου, και άλλα γνωστά στοιχεία ελέγχου παρέχουν τυπικές δυνατότητες επεξεργασίας για σελίδες Web Forms. Ωστόσο, μπορούν να προγραμματιστούν και με κώδικα που τρέχει στον server, και όχι τον πελάτη.

Σε αυτή την ενότητα θα προστεθούν τα ακόλουθα στοιχεία ελέγχου: ένα κουμπί (button), ένα πλαίσιο κειμένου (Text Box) και μία ετικέτα (label)[8]. Επίσης, θα γραφτεί κώδικας για το χειρισμό του γεγονότος Click στο κουμπί. Για να προστεθούν τα στοιχεία ελέγχου στη σελίδα:

- Επιλέγεται η καρτέλα Design για τη μετάβαση σε προβολή σχεδίασης.
- Τοποθετείται το σημείο εισαγωγής στο τέλος του Welcome to Visual Web Developer κειμένου και με ENTER πέντε ή περισσότερες φορές μπορεί να υπάρξει περισσότερος χώρος στο πλαίσιο div.
- Στην εργαλειοθήκη (toolbox), αν δεν φαίνονται τα στοιχεία της ομάδας Standard, πρέπει να επεκταθεί.
- Με σύρσιμο, μπορεί να τοποθετηθεί ένα στοιχείο TextBox στο μέσο του πλαισίου div που περιέχει το κείμενο στην πρώτη γραμμή.
- Με σύρσιμο, μπορεί να τοποθετηθεί ένα στοιχείο Button στη σελίδα, δεξιά από το πλαίσιο κειμένου.
- Με σύρσιμο, μπορεί να τοποθετηθεί ένα στοιχείο Label στη σελίδα σε μια γραμμή κάτω από το στοιχείο Button.
- Πάνω από το TextBox εισάγεται το κείμενο “Enter your name:”.

Αυτό το στατικό κείμενο HTML είναι η λεζάντα για τον στοιχείο ελέγχου κειμένου. Στην ίδια σελίδα μπορούν να αναμιχθούν στατική HTML και server controls.

[8] https://msdn.microsoft.com/library/system.web.ui.webcontrols.label.aspx

Συμπέρασμα

Στο συγκεκριμένο κεφάλαιο παραθέτουμε επιπλέον στοιχεία για τις εφαρμογές της ASP.NET web forms και κατανοούμε τους λόγους που την επιλέγουν αρκετοί προγραμματιστές.

Κεφάλαιο 5
Web Pages

Στο κεφάλαιο 5 αναλύουμε τα βασικά χαρακτηριστικά του web matrix καθώς και τα βήματα της δημιουργίας μιας απλής ιστοσελίδας με το Web Pages. Σημαντική ενότητα του κεφαλαίου είναι και η σύγκριση της ASP.NET με την PHP.

5.1 WebMatrix

Το WebMatrix είναι ένα εργαλείο που ενσωματώνει έναν επεξεργαστή ιστοσελίδων, ένα βοηθητικό πρόγραμμα βάσης δεδομένων, έναν web server για τη δοκιμή σελίδων και τα χαρακτηριστικά για τη δημοσίευση της ιστοσελίδας στο Internet. Το WebMatrix είναι δωρεάν, και είναι εύκολο στην εγκατάσταση και στη χρήση του. (Λειτουργεί, επίσης, για απλές HTML σελίδες, καθώς επίσης και για άλλες τεχνολογίες, όπως η PHP.)

Εκτός από το WebMatrix, μπορούν να δημιουργηθούν σελίδες χρησιμοποιώντας ένα πρόγραμμα επεξεργασίας κειμένου, για παράδειγμα, και ο έλεγχός τους να γίνει με τη χρήση ενός web server στον οποίο υπάρχει πρόσβαση. Ωστόσο, το WebMatrix καθιστά τα πάντα πολύ πιο εύκολα, οπότε στην παρουσίαση αυτού του κεφαλαίου θα χρησιμοποιηθεί το WebMatrix.

5.2 Τελικά είναι καλύτερη η PHP ή ASP.NET?

Για να αποφασίσει κάποιος αν είναι καλύτερη η PHP ή η ASP.NET θα πρέπει να ξεκαθαρίσει το τι ακριβώς θελει να επιτύχει, ποιος είναι ο στόχος του δηλαδή και για ποιον λόγο χρειάζεται μια από αυτές τις γλώσσες. Παρακάτω ακολουθεί μια λίστα όπου γίνεται σύγκριση σε κάποια πεδία μεταξύ της PHP και της ASP.NET.

- Απόδοση και ταχύτητα : Αρχικά**,** μια κοινή εσφαλμένη αντίληψη σχετικά με την ταχύτητα και την απόδοση του ιστότοπου είναι ότι η γλώσσα που επιλέγει κάποιος να κωδικοποιήσει καθορίζει την συνολική απόδοση του συστήματος. Στην πραγματικότητα ωστόσο υπάρχει πολύ μικρή διαφορά μεταξύ της απόδοσης ενός ιστότοπου εκτέλεσης με PHP και ενός ιστότοπου εκτέλεσης με

ASP.NET. Γενικά επικρατεί η άποψη οτι η ASP.NET είναι πιο κομψή γλώσσα από την PHP.

- Η ASP.NET δεν επιτρέπει πολλές παρατυπίες στον κώδικα της, ενώ στην PHP αυτό μπορεί να συμβεί(τεχνικά η PHP είναι μια πιο εύκολη γλώσσα για αρχάριους). Τέλος, πολλοί <<ειδικοί>> στον κόσμο του προγραμματισμού προσπαθούν να υποβιβάσουν την PHP, παραβλέποντας το γεγονός ότι αφαρμογές όπως MailChimp και το Facebook είναι γραμμένες σε PHP, γεγονός που αποδεικνύει τις δυνατότητες και την δημοφιλία της.
- Επεκτασιμότητα: Τόσο η ASP.NET όσο και η PHP είναι εξαιρετικά επεκτάσιμες γλώσσες. Αυτό που έχει μεγαλύτερη σημασία για την επεκτασιμότητα από τη γλώσσα που επιλέγεται είναι το <<ταλέντο>> ανάπτυξης που έχει ο προγραμματιστής.
- Υποστήριξη: Επειδή η PHP είναι ανοιχτή πηγή, η ομάδα των προγραμματιστών της είναι πολύ μεγαλύτερη από αυτήν της ASP.NET. Δηλαδή, και οι 2 μπορούν να υπερηφανεύονται για τα live forums που δημοσιεύουν τακτικά στο διαδίκτυο, οπότε αν ψάχνετε για απαντήσεις σε προβλήματα, είναι πιθανό να βρείτε και τις δύο κοινότητες χρήσιμες. Ενώ η κοινότητα της ASP.NET αποτελείται από εξειδικευμένους προγραμματιστές, υπάρχουν αρκετά λιγότεροι διαθέσιμοι σε θέματα υποστήριξης που είναι πρόθυμοι και ικανοί να δημοσιεύσουν φόρουμ και να απαντήσουν σε ερωτήσεις σχετικά με τις προκλήσεις της ASP.NET. Σε αντίθεση με αυτό, η PHP είναι μια τόσο ευρέως χρησιμοποιούμενη γλώσσα, όπου υπάρχουν πολλοί φιλικοί προγραμματιστές που δραστηριοποιούνται σε διαφορετικά φόρουμ που είναι περισσότερο πρόθυμοι να προσφέρουν δωρεάν συμβουλές και καθοδήγηση σε όσους το ζητούν. Η μεγαλύτερη διαφορά εδώ είναι ότι ενώ θα είστε πιθανότατα σε θέση να βρείτε απαντήσεις στις ερωτήσεις σας και στις δύο κοινότητες, θα πάρετε σχεδόν με βεβαιότητα αυτές τις απαντήσεις πιο γρήγορα όταν απευθύνεστε σε φόρουμ της PHP.
- Κόστος: Αυτή είναι η μόνη περιοχή στην οποία η PHP κατέχει ένα ξεχωριστό πλεονέκτημα έναντι της ASP.NET. Η PHP είναι ανοικτού κώδικα, και επομένως, εντελώς δωρεάν, ενώ η ASP.NET ανήκει στη Microsoft και έρχεται με αμοιβή web hosting.Τέλος,αξίζει επίσης να σημειωθεί ότι η PHP μπορεί να

χρησιμοποιηθεί σε μηχανές Mac, Windows ή Linux, ενώ η ASP.NET προορίζεται μόνο για υπολογιστές με Windows.

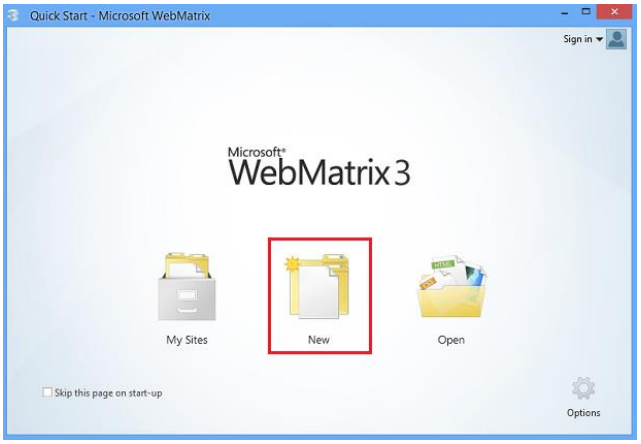
- Μέσος χρόνος ανάπτυξης: Για εφαρμογές μικρότερου μεγέθους πιο χρήσιμη είναι η PHP. Αν στόχος είναι η ανάπτυξη εφαρμογής μεγάλης κλίμακας, τότε θα πρέπει να επιλέξει κάποιος την ASP.NET & ASP.NET MVC.
- Πλαίσια: Η PHP είναι καταλληλότερο πλαίσιο για μικρές εφαρμογές , όπως ήδη αναφέρθηκε. Η PHP τρέχει χωρίς πολλές προδιαγραφές/απαιτήσεις. Η Microsoft ASP.NET είναι ένα βιομηχανικό ισχυρό πλαίσιο, το οποίο περιέχει οτιδήποτε αναζητάει ένας έμπειρος χρήστης καθώς περιέχει ένα πολύ πλούσιο περιβάλλον χαρακτηριστικών.

5.3 Δημιουργία μιας απλής ιστοσελίδας με το Web Pages

5.3.1 Δημιουργία web site και σελίδας

Εφόσον είναι εγκατεστημένο το WebMatrix, μπορεί να ξεκινήσει στα Windows, από το μενού Έναρξη, επιλέγοντας Microsoft WebMatrix.

Για να δημιουργηθεί ένας κενός ιστότοπος και να προστεθεί μια σελίδα σε αυτόν, κλικ στο New (**Error! Reference source not found.**18).



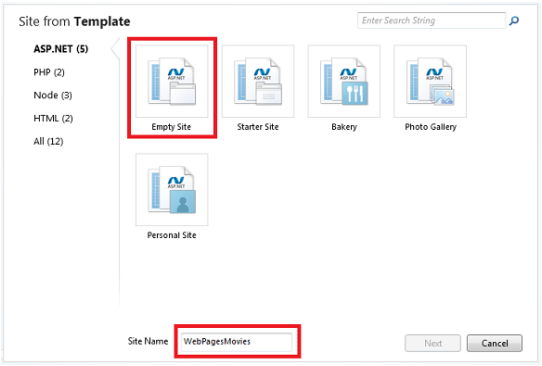
Εικόνα 15: Η οθόνη εκκίνησης του Microsoft Web Matrix.

Τα πρότυπα είναι έτοιμα αρχεία και σελίδες για διάφορους τύπους δικτυακών τόπων. Για να εμφανιστούν όλα τα πρότυπα που είναι διαθέσιμα από προεπιλογή, κλικ στην επιλογή Template Gallery (**Error! Reference source not found.**19).

Στο παράθυρο Quick Start, με την επιλογή Empty Site από το γκρουπ του ASP.NET δημιουργείται ένα νέο κενό site, στο οποίο μπορεί να δοθεί μια ονομασία, π.χ.

"WebPagesMovies" (**Error! Reference source not found.**0). Έπειτα, κλικ στο κουμπί Next.

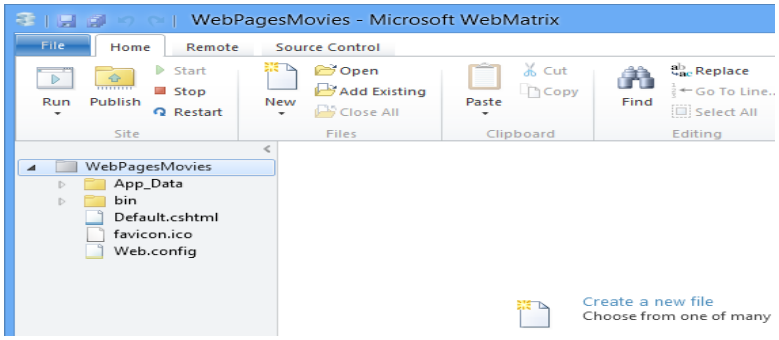
Εικόνα 169: Πρότυπα ιστοσελίδων στο Microsoft Web Matrix.



Εικόνα 170: Δημιουργία νέο κενού site στο Microsoft Web Matrix.

Εάν έχει υπάρξει σύνδεση στο λογαριασμό Microsoft, δίνεται η ευκαιρία δημιουργίας της ιστοσελίδας στο Azure (διαδικτυακή βάση δεδομένων).

Για την εξοικείωση με το WebMatrix και το ASP.NET Web Pages, θα δημιουργηθεί μια απλή σελίδα. Στον επιλογέα χώρου εργασίας, ο χώρος Files επιτρέπει την εργασία με αρχεία και φακέλους. Το αριστερό τμήμα του παραθύρου δείχνει τη δομή των αρχείων του site (**Error! Reference source not found.**21). Η κορδέλα αλλάζει για να δείξει τις λειτουργίες που σχετίζονται με αρχεία.



Εικόνα 18: Το περιβάλλον εργασίας με site του Web Matrix.

- Για να δημιουργηθεί ένα νέο αρχείο επιλέγουμε το New και εμφανίζεται η επιθυμητή επιλογή New File . Το WebMatrix εμφανίζει μια λίστα των τύπων αρχείων. Μπορούμε να επιλέξουμε CSHTML.Οι σελίδες τύπου CSHTML είναι οι σελίδες του ASP.NET Web Pages.Μόλις πατηθεί το OK, το WebMatrix δημιουργεί την σελίδα και την ανοίγει στον editor.
- Στον χώρο εργασίας Files, με δεξί-κλικ στη σελίδα.cshtml και κλικ στο Launch in browser, μπορεί να ελεγχθεί η εκτέλεση της σελίδας.Το WebMatrix ξεκινά έναν built-in web server (IIS Express) που μπορεί αν χρησιμοποιηθεί στην δοκιμή σελίδων στον υπολογιστή. (Χωρίς το IIS Express στο WebMatrix, θα έπρεπε να δημοσιευτεί κάπου η σελίδα προτού δοκιμαστεί.) Η σελίδα εμφανίζεται στον προεπιλεγμένο browser .

Να σημειωθεί ότι όταν δοκιμάζεται η σελίδα σε WebMatrix, το URL στον browser είναι κάπως έτσι: http://localhost:33651/HelloWorld.cshtml. Το όνομα localhost αναφέρεται στον τοπικό server, πράγμα που σημαίνει ότι η σελίδα εξυπηρετείται από ένα web server που είναι στον υπολογιστή. Όπως προαναφέρθηκε, το WebMatrix περιλαμβάνει ένα πρόγραμμα που ονομάζεται web server IIS Express που εκτελείται.

Συμπέρασμα

Γενικά, η ASP.NET παράγει δυναμικές ιστοσελίδες. Στο παράδειγμα μας, η σελίδα που είδαμε δημιουργήθηκε με HTML. Μπορεί επίσης να περιέχει κώδικα που μπορεί να εκτελέσει όλα τα είδη των λειτουργιών. Όταν καλείται η σελίδα .cshtml στον browser, η ASP.NET επεξεργάζεται τη σελίδα, ενώ είναι ακόμα υπό την κατοχή του web server. Τέλος, είδαμε πως ξεκάθαρη σύγκριση μεταξύ της PHP και της ASP.NET δεν μπορεί να γίνει διότι κάθε μια από τις γλώσσες έχει τα πλεονεκτήματα και τα μεινοκτήματά της.

Κεφάλαιο 6
ASP.NET Razor

Στο κεφάλαιο 6 θα μιλήσουμε για την server side γλώσσα σήμανσης Razor, για τον τρόπο που συντάσσεται, για τα Razor helpers καθώς και για διάφορα ερωτήματα σχετικά με την Razor και τις απαντήσεις τους.

6.1 Τι είναι η Razor

Η Razor δεν είναι μια γλώσσα προγραμματισμού. Πρόκειται για μια server side γλώσσα σήμανσης, η σύνταξη της οποίας επιτρέπει την ενσωμάτωση server-based κώδικα (Visual Basic and C#) σε ιστοσελίδες.

Ο server-based κώδικας μπορεί να δημιουργήσει δυναμικό web περιεχόμενο σε πραγματικό χρόνο (on the fly), ενώ μια ιστοσελίδα είναι γραμμένη στο πρόγραμμα περιήγησης. Όταν μια ιστοσελίδα καλείται, ο διακομιστής εκτελεί τον server-based κώδικα στο εσωτερικό της σελίδας πριν επιστρέψει η σελίδα στο πρόγραμμα περιήγησης. Εξαιτίας της εκτέλεσής του στον server, ο κώδικας μπορεί να φέρει σε πέρας πολύπλοκες εργασίες, πχ. την πρόσβαση σε βάσεις δεδομένων.

Η Razor είναι βασισμένη στην ASP.NET, και σχεδιάστηκε για την δημιουργία web εφαρμογών. Έχει την ισχύ της παραδοσιακής σήμανσης ASP.NET, όμως είναι ευκολότερη στη χρήση και στην εκμάθηση.

6.2 Σύνταξη της Razor

Η Razor χρησιμοποιεί μια σύνταξη πολύ παρόμοια με την PHP και την Classic ASP.

Ένα παράδειγμα με την γλώσσα Razor ,που όταν εκτελεστεί θα εμφανίσει σε μορφή λίστας (<li>)κουκκίδων (ul) τους αριθμούς από το 0 μέχρι το 9.

Razor:

```
<u>
@for (int i=0; i<10; i++) {
<li>@i</li>
}
```

6.2.1 Βασικοί Συντακτικοί Κανόνες Razor για C#

Οι βασικοί κανόνες σύνταξης της Razor για την γλώσσα C# είναι οι εξής:

- Τα blocks κώδικα Razor περικλείονται σε @{ ... }
- Οι ενσωματωμένες εκφράσεις (μεταβλητές και συναρτήσεις) αρχίζουν με @
- Οι δηλώσεις κώδικα τελειώνουν με άνω τελεία
- Οι μεταβλητές δηλώνονται με τη λέξη-κλειδί var
- Τα Strings περικλείονται σε εισαγωγικά
- Ο κώδικας C# κάνει διάκριση μεταξύ πεζών-κεφαλαίων (case-sensitive).

6.2.2 Βασικοί Συντακτικοί Κανόνες Razor για Visual Basic

Οι βασικοί κανόνες σύνταξης της Razor για την γλώσσα VB είναι οι εξής:

- Τα Razor code blocks περικλείονται σε @Code ... End Code
- Οι ενσωματωμένες εκφράσεις (μεταβλητές και συναρτήσεις) αρχίζουν με @
- Οι μεταβλητές δηλώνονται με την λέξη-κλειδί Dim
- Τα Strings περικλείονται σε εισαγωγικά
- Ο κώδικας VB δεν κάνει διάκριση μεταξύ πεζών και κεφαλαίων
- Οι VB φάκελοι έχουν την .vbhtml επέκταση

6.3 Razor Helpers

Οι βοηθοί (helpers) της ASP.NET είναι στοιχεία στα οποία μπορεί να υπάρχει πρόσβαση από ενιαίες γραμμές του κώδικα Razor. Μπορούν είτε να φτιαχτούν νέοι βοηθοί χρησιμοποιώντας τη σύνταξη Razor, ή να χρησιμοποιηθούν οι προκατασκευασμένοι helpers της ASP.NET.

Οι λειτουργίες μερικών χρήσιμων Razor βοηθών είναι:

- Web Graphics
- Google Analytics
- Facebook Integration
- Twitter Integration
- Sending Email
- Validation

6.4 Απαντήσεις σε ερωτήματα για την Razor

Παρακάτω ακολουθεί μια σειρά από ερωτήσεις και απαντήσεις σε σημαντικά θέματα που αφορούν την Razor.

6.4.1 Είναι απαραίτητο το Web Matrix προκειμένου να δουλέψω με τις Web Pages?

Η απάντηση σε αυτό το ερώτημα είναι ξεκάθαρη. Το Web Matrix δεν αποτελεί ολοκληρωμένο περιβάλλον ανάπτυξης για ιστοσελίδες ASP.NET . Φρόνιμο είναι να χρησιμοποιηθεί το Visual Studio.

Αν δεν επιθυμεί κάποιος να χρησιμοποιήσει το Visual Studio, μπορεί να χρησιμοποιήσει μεμονωμένα προϊόντα της Microsoft Web Installer Platform, τα οποία είναι τα εξής

- Microsoft .NET Framework 4.5
- ASP.NET MVC 5 (εγκαθιστεί άψογα το ASP.NET Web Pages Framework)
- IIS Express (Web server)
- Microsoft SQL Server Compact 4.0 (βάση δεδομένων)

Η διαχείριση SQL Server Compact βάσης δεδομένων (αρχεία .sdf) χωρίς την χρήση κάποιου εργαλείου είναι δύσκολη . Το Visual Studio παρέχει εργαλεία για την διαχείριση .sdf βάσεων δεδομένων. Μπορεί κάποιος επίσης να τρέξει εντολές SQL σε κώδικα για να εκτελέσει πολλές εργασίες διαχειίρισης SQL Server.

Για να δοκιμάσει κάποιος τις σελίδες .cshtml χωρίς να χρησιμοποιήσει ένα ολοκλρωμένο περιβάλλον ανάπτυξης (IDE), μπορεί να φορτώσει τις σελίδες σε έναν live server.

6.4.2 Μπορώ να χρησιμοποιήσω τα στοιχεία ελέγχου ASP. NET Web Forms σε μια ιστοσελίδα?

Όχι. Τα στοιχεία ελέγχου των Web Forms, όπως το κουμπί ελέγχου, τα κουμπιά επαλήθευσης δουλέυουν μόνο σε Web Forms Pages (.aspx αρχεία). Αυτά τα κουμπιά απαιτούν την Web Form page platform.

6.4.3 Μπορώ να χρησιμοποιήσω το WebSecurity helper για να υποστηρίξω το login?

Οι παροχείς ασφάλειας , οι οποίοι είναι κομμάτι της ASP.NET πιθανόν να το έχει χρησιμοποιήσει κάποιος δουλεύοντας με τις Web Forms) είναι διαθέσιμοι. Για παράδειγμα, μπορεί κάποιος να χρησιμοποιήσει τις φόρμες αυθεντικοποίησης των ASP.NET Web Pages όπως ακριβώς και στις Web Forms.

6.6.4 Μπορώ να δουλέψω σε μια σελίδα ASP.NET Web Pages χωρίς να χρησιμοποιήσω το WebMatrix?

Η απάντηση σε αυτήν την ερώτηση είναι καταφατική. Μπορεί κάποιος να αντιγράψει χειροκίνητα, μη αυτόματα δηλαδή, αρχεία ιστότοπου στομν διακομιστή (τυπικά χρησιμοποιώντας FTP). Όταν εκτελεί κάποιος ένα χειροκίνητο αντίγραφο, τότε πρέπει να αντιγράψει τα αρχεία που υποστηρίζουν το SQL Server Compact (την βάση δεδομένων).

6.6.5 Οι ASP.NET Web Pages υποστηρίζουν την HTML5?

Φυσικά. Οι σελίδες που σημιουργούνται με ASP.NET Web Pages (. Cshtml ή .vbhtml σελίδες) είναι HTML σελίδες στην πραγματικότητα που επίσης περιέχουν κώδικα, ο οποίος τρέχει στον server, πριν ακόμα η σελίδα αποδοθεί. Για όσο διάστημα ο browser του χρήστη υποστηρίζει την HTML5, για τόσο μπορούν να χρησιμοποιηθούν τα HTML5 στοιχεία σε μια .cshtml ή .vbhtml σελίδα.

Συμπέρασμα

Επιγραμματικά, τα βασικά σημεία της Razor είναι τα εξής:

- Η Razor είναι σήμανση σύνταξης για την προσθήκη server-based κώδικα σε ιστοσελίδες
- Η Razor έχει την ισχύ της παραδοσιακής ASP.NET σήμανσης, αλλά είναι ευκολότερη στην εκμάθηση και στην χρήση
- Η Razor είναι μια server side σήμανση σύνταξης σαν την ASP και την PHP
- Η Razor υποστηρίζει τις γλώσσες προγραμματισμού C# και Visual Basic.

Κεφάλαιο 7
Single Page Apps

Στο κεφάλαιο 7 θα μιλήσουμε για τις εφαρμογές Single Page με ASP.NET για την Single Page εφαρμογή MovieSPA, για τη δημιουργία του Project σε Visual Studio, για τα μοντέλα MVC και MVVM, για τη δημιουργία του Web Client με το Knockout.js, ,για τη δημιουργία του μοντέλου προβολής (View Model)για τα Data Bindings, για την επεξεργασία εγγραφών και για τη δημιουργία Web Client με το Ember.

7.1 Εφαρμογές Single-Page με ASP.NET

Οι Single-Page εφαρμογές (SPAs) είναι web εφαρμογές που φορτώνουν μια απλή HTML σελίδα και δυναμικά ενημερώνουν αυτή τη σελίδα καθώς ο χρήστης αλληλεπιδρά με την εφαρμογή[9].

Οι SPAs χρησιμοποιούν την AJAX και την HTML5 για να δημιουργήσουν γρήγορες και ανταποκρίσιμες web εφαρμογές, χωρίς τις συνεχής επαναφορτώσεις της σελίδας. Ωστόσο, αυτό σημαίνει ότι ένα μεγάλο μέρος της εργασίας γίνεται από την πλευρά του client, σε JavaScript. Για τον παραδοσιακό προγραμματιστή της ASP.NET, ίσως είναι δύσκολο να γίνει το άλμα. Ευτυχώς, υπάρχουν πολλά open source πλαίσια JavaScript που κάνουν ευκολότερη τη δημιουργία των SPAs.

Σε αυτό το κεφάλαιο θα γίνει η παρουσίαση των Single Page εφαρμογών, μέσω της δημιουργίας μιας απλής SPA. Κατά την περιήγηση, θα παρουσιαστούν κάποιες θεμελιώδεις αρχές για την δημιουργία SPAs, συμπεριλαμβανομένων των μοτίβων Model-View-Controller (MVC) και Model-View-ViewModel (MVVM), του data binding και της δρομολόγησης (routing).

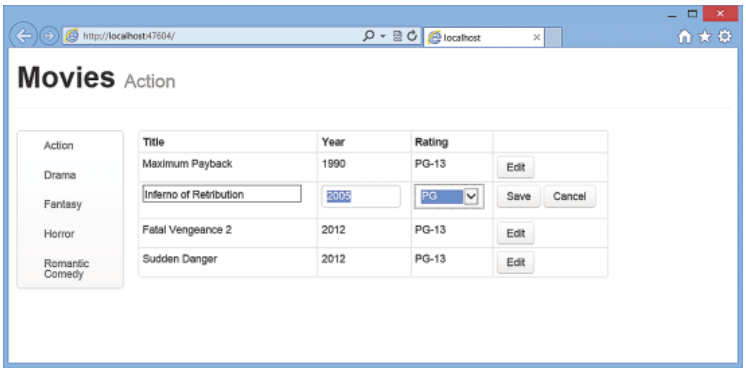
7.2 Η Single Page εφαρμογή MovieSPA

Η εφαρμογή – δείγμα που θα δημιουργηθεί είναι μια απλή βάση δεδομένων ταινιών, όπως φαίνεται στην Εικόνα 22. Η αριστερή στήλη της σελίδας εμφανίζει μια λίστα ειδών. Κάνοντας κλικ σε ένα είδος εμφανίζεται μια λίστα με ταινίες του είδους αυτού. Πατώντας το κουμπί Edit δίπλα σε μια καταχώρηση επιτρέπει την αλλαγή αυτής της

[9] https://msdn.microsoft.com/en-us/magazine/dn463786.aspx

καταχώρησης. Αφού κάνει αλλαγές, ο προγραμματιστής μπορεί να κάνει κλικ στο κουμπί Save για να υποβάλει την ενημερωμένη έκδοση στον server, ή να πατήσει το κουμπί Cancel για να επαναφέρει τις αλλαγές.

Υπάρχουν δύο διαφορετικές εκδόσεις του app, μία χρησιμοποιώντας τη βιβλιοθήκη Knockout.js και η άλλη με τη χρήση της βιβλιοθήκης Ember.js . Αυτές οι δύο βιβλιοθήκες έχουν διαφορετικές προσεγγίσεις, γι αυτό είναι χρήσιμη η σύγκρισή τους. Και στις δυο περιπτώσεις, η client app ήταν λιγότερη από 150 γραμμές JavaScript. Για την πλευρά του server χρησιμοποιείται το ASP.NET Web API για να εξυπηρετεί το JSON στον client10.



Εικόνα 19: Η Single-Page εφαρμογή MovieSPA.

7.3 Θεωρητικό Υπόβαθρο

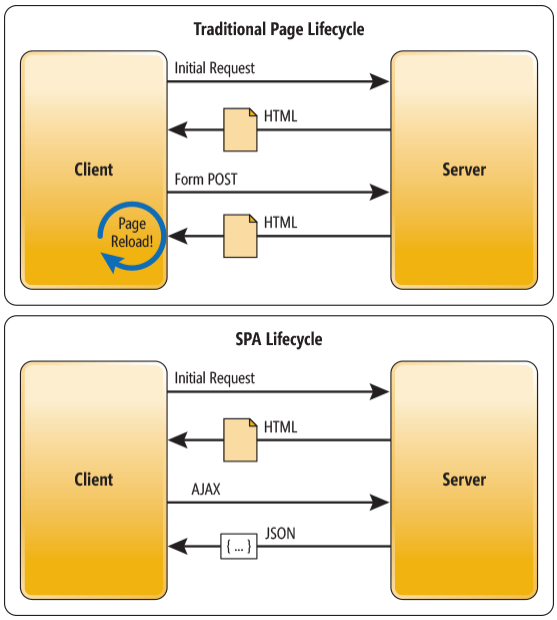
Σε μια παραδοσιακή Web εφαρμογή, κάθε φορά που η εφαρμογή καλεί τον server, εκείνος δημιουργεί μια νέα σελίδα HTML. Αυτό πυροδοτεί την ανανέωση της σελίδας στον browser. Ο κύκλος ζωής σελίδας που μόλις περιγράφθηκε είναι κοινός για εφαρμογές Web Forms (βλ. Κεφάλαιο 3) ή PHP εφαρμογές.

Σε μια SPA, μετά την φόρτωση της πρώτης σελίδας, όλη η αλληλεπίδραση με τον server γίνεται μέσω κλήσεων AJAX. Αυτές οι κλήσεις AJAX επιστρέφουν δεδομένα —όχι σήμανση— συνήθως σε μορφή JSON. Η εφαρμογή χρησιμοποιεί τα δεδομένα της JSON για να ενημερώσει δυναμικά την σελίδα, χωρίς να επαναφορτώσει την σελίδα. Η **Error! Reference source not found.**23 απεικονίζει τη διαφορά μεταξύ των δύο προσεγγίσεων.

Ένα πλεονέκτημα των SPAs είναι προφανές: Οι εφαρμογές είναι περισσότερο ρευστές και να ανταποκρίσιμες, χωρίς την ενοχλητική επίδραση της μεταφόρτωσης

[10] github.com/MikeWasson/MoviesSPA

και την εκ νέου απόδοσης της σελίδας. Ένα άλλο όφελος που είναι λιγότερο προφανές αφορά το πώς ο προγραμματιστής θα δημιουργήσει μια web εφαρμογή. Στέλνοντας τα δεδομένα της εφαρμογής ως JSON δημιουργεί ένα διαχωρισμό μεταξύ της παρουσίασης (σήμανση της HTML) και της λογική της εφαρμογής (αιτήματα AJAX και αποκρίσεις JSON).



Εικόνα 20:Ο παραδοσιακός κύκλος ζωής της σελίδας και ο κύκλος ζωής της SPA.

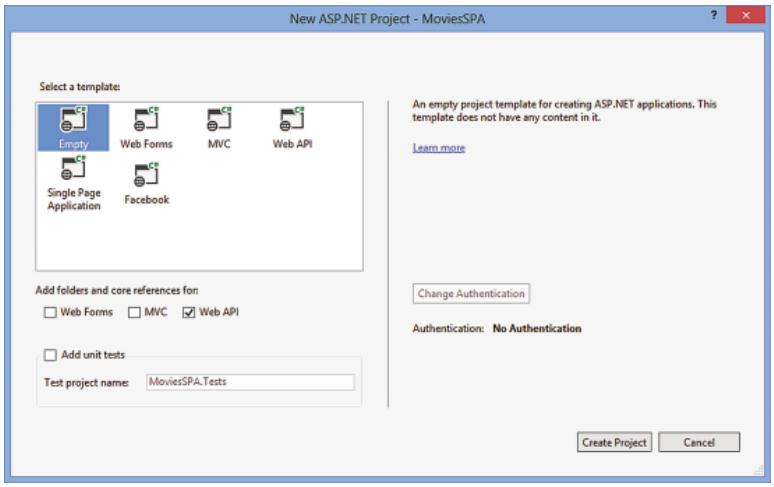
Ο διαχωρισμός κάνει ευκολότερο τον σχεδιασμό και την ανάπτυξη κάθε επιπέδου. Σε μια καλό-σχεδιασμένη SPA, μπορούν να γίνουν αλλαγές στην σήμανση της HTML χωρίς να πειραχτεί ο κώδικας που υλοποιεί την λογική της εφαρμογής (τουλάχιστον, αυτό είναι το ιδανικό). Αυτό θα γίνει περισσότερο κατανοητό στην ενότητα που αφορά το data binding.

Σε μια καθαρή SPA, όλη η αλληλεπίδραση UI συμβαίνει στον client, μέσω της JavaScript και της CSS. Μετά τη φόρτωση της αρχικής σελίδας , ο διακομιστής λειτουργεί καθαρά ως ένα στρώμα υπηρεσίας. Ο πελάτης απλώς χρειάζεται να ξέρει ποιά HTTP αιτήματα να στείλει. Δεν τον ενδιαφέρει το πώς ο server υλοποιεί τα πράγματα στο back-end.

Με την αρχιτεκτονική αυτή, ο πελάτης είναι ανεξάρτητος από την υπηρεσία (και το αντίστροφο). Μπορεί να αντικατασταθεί ολόκληρο το back end το οποίο τρέχει την υπηρεσία, και εφ 'όσον δεν αλλάξει το API, δεν θα επηρεαστεί ο πελάτης. Το αντίστροφο είναι επίσης αληθές: μπορεί να αντικατασταθεί το σύνολο της εφαρμογής του πελάτη, χωρίς να αλλάξει το στρώμα υπηρεσίας. Για παράδειγμα, μπορεί να γραφτεί ένας νέος mobile client που χρησιμοποιεί την ίδια υπηρεσία.

7.4 Δημιουργία του Project σε Visual Studio

Το Visual Studio 2013 έχει έναν τύπο έργου ASP.NET Web Application. Το πρόγραμμα wizard επιτρέπει την επιλογή των συστατικών του ASP.NET που θα συμπεριληφθούν στη νέα εφαρμογή. Ξεκινώντας με το πρότυπο Empty, στην συνέχεια προστέθηκε το ASP.NET Web API στο project επιλέγοντας το Web API κάτω από το "Add folders and core references for:" όπως φαίνεται στην **Error! Reference source not found.**24 .



Εικόνα 21:Δημιουργία νέου ASP.NET προγράμματος σε Visual Studio 2013.

Το νέο έργο έχει όλες τις βιβλιοθήκες που χρειάζονται για το Web API, συν κάποιους κώδικες διαμόρφωσης του Web API.

7.4.1 Δημιουργία του Service Layer

Χρησιμοποιήθηκε το ASP.NET Web API για την δημιουργία ενός απλού REST (representational state transfer) API για την εφαρμογή. Το ASP.NET Web API είναι ένα πλαίσιο που διευκολύνει την δημιουργία HTTP υπηρεσιών που φθάνουν σε μεγάλη κλίμακα clients, όπως οι browsers και οι κινητές συσκευές11.

Κατ 'αρχήν, δημιουργήθηκε μια κλάση Movie που αντιπροσωπεύει μια ταινία. Αυτή η κλάση κάνει δύο πράγματα:

- Λέει στο Entity Framework (EF) πώς να δημιουργήσει τους πίνακες της βάσης δεδομένων για την αποθήκευση των δεδομένων της ταινίας.
- Λέει στο Web API πώς να διαμορφώσει το ωφέλιμο φορτίο της JSON.

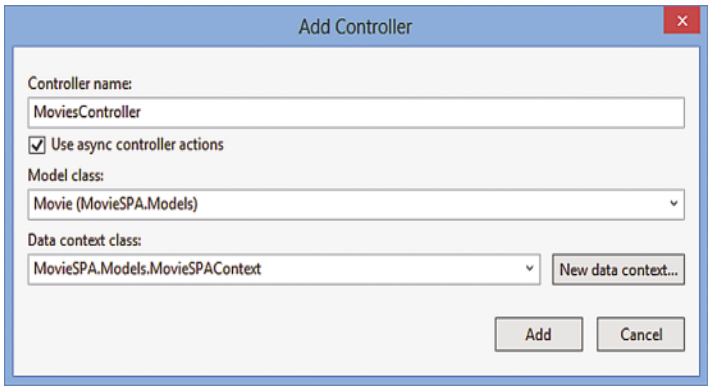
Δεν είναι απαραίτητο να χρησιμοποιηθεί το ίδιο μοντέλο και για τα δύο.

[11] http://www.asp.net/web-api

Η **Error! Reference source not found.**25 δείχνει τον Add Controller wizard. Το controller ονομάστηκε MoviesController. Το όνομα έχει σημασία, επειδή τα URIs για το REST API είναι βασισμένα στο όνομα του controller. Έγινε επιλογή της κλάσης Movie για το μοντέλο και του "New data context" για την δημιουργία ενός νέου πλαίσιο δεδομένων EF.

Ο wizard προσθέτει δυο αρχεία:

- Το MoviesController.*cs* καθορίζει το Web API controller που υλοποιεί το REST API για την εφαρμογή.
- Το MovieSPAContext.*cs* παρέχει μεθόδους για ερωτήματα στην υποκείμενη βάση δεδομένων.



Εικόνα 22: Ο Add Controller Wizard.

7.4.2 Δημιουργία του Web Client

Μέχρι στιγμής, έχει δημιουργηθεί ένα REST API.

Τώρα πρέπει να γραφτεί μια εφαρμογή πελάτη. Η βασική ροή εργασίας είναι:

- Το UI ενεργοποιεί ένα αίτημα AJAX
- Ανανέωση της HTML για να εμφανιστεί η απάντηση του payload
- Χειρισμός λαθών της AJAX

Όλα αυτά μπορούν να κωδικοποιηθούν με το χέρι. Για παράδειγμα, εδώ είναι ένας κώδικας jQuery που δημιουργεί μια λίστα με τίτλους ταινιών:

```
$.getJSON(url)
.done(function (data) {
// On success, "data" contains a list of movies
var ul = $("<ul></ul>")
$.each(data, function (key, item) {
```

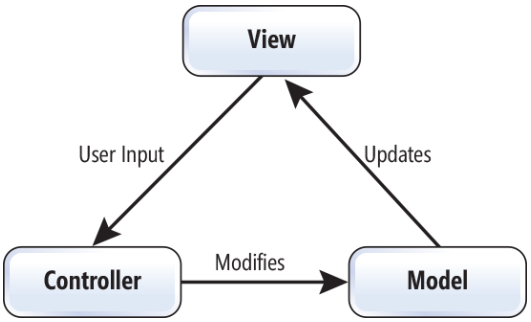
```
// Add a list item
$('<li>', { text: item.Title }).appendTo(ul);
});
$('#movies').html(ul);
});
```

Ο κώδικας αυτός έχει κάποια προβλήματα. Συνδυάζει τη λογική της εφαρμογής με τη λογική παρουσίαση, και είναι στενά συνδεδεμένος με την HTML. Επίσης, είναι κουραστική η συγγραφή του. Η λύση είναι να κατασκευαστεί πάνω σε ένα πλαίσιο JavaScript. Ευτυχώς, μπορεί να γίνει επιλογή από πολλά πλαίσια ανοικτού κώδικα JavaScript. Μερικά από τα πιο δημοφιλή περιλαμβάνουν το Backbone, το Angular, το Ember, το Knockout, το Dojo και το JavaScriptMVC. Οι περισσότεροι χρησιμοποιούν κάποια παραλλαγή των προτύπων MVC ή MVVM, γι 'αυτό κρίνεται σκόπιμο να παρουσιαστούν αυτά τα πλαίσια στη συνέχεια.

7.5 Τα μοντέλα MVC και MVVM

Το μοντέλο MVC χρονολογείται κατά τη δεκαετία του 1980 και στις απαρχές των γραφικών διεπαφών χρήστη. Ο στόχος του MVC είναι να παράγει του κώδικα σε τρεις ξεχωριστές αρμοδιότητες, όπως φαίνεται στην **Error! Reference source not found.**26. Αυτές είναι οι εξής:

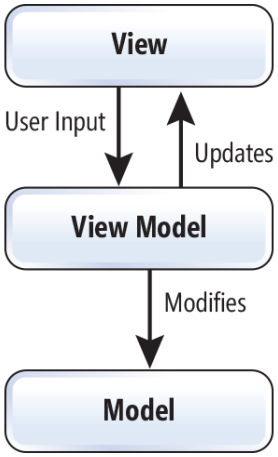
- Το μοντέλο (model) αναπαριστά τον τομέα των δεδομένων και την επιχειρησιακή λογική
- Η προβολή (view) εμφανίζει το μοντέλο.
- Ο ελεγκτής (controller) λαμβάνει είσοδο από τον χρήστη και ενημερώνει το μοντέλο.



Εικόνα 23: Το μοντέλο MVC

Μια πιο πρόσφατη παραλλαγή του MVC είναι το πρότυπο MVVM (**Error! Reference source not found.**27). Στο MVVM:

- Το μοντέλο (model) εξακολουθεί να αντιπροσωπεύει τα δεδομένα του τομέα.
- Το μοντέλο *προβολής* (view model) είναι μια αφηρημένη αναπαράσταση της προβολής.
- Η προβολή (view) εμφανίζει το μοντέλο προβολή και στέλνει την είσοδο του χρήστη στο μοντέλο προβολής.



Εικόνα 24: Το μοντέλο MVVM.

Σε ένα πλαίσιο JavaScript MVVM, η προβολή είναι σήμανση και το μοντέλο προβολής είναι κώδικας.

Το MVC έχει πολλές παραλλαγές, και η βιβλιογραφικές αναφορές σχετικά με το MVC είναι συχνά ασαφείς και αντιφατικές. Ίσως αυτό δεν είναι περίεργο για ένα πρότυπο σχεδιασμού που ξεκίνησε με Smalltalk-76 και εξακολουθεί να χρησιμοποιείται στις σύγχρονες Web Apps. Έτσι, ακόμα κι αν είναι καλό να υπάρχει επίγνωση της θεωρίας, το κύριο πράγμα είναι η κατανόηση του συγκεκριμένου πλαισίου MVC που χρησιμοποιείται.

7.6 Δημιουργία του Web Client με το Knockout.js

Για την πρώτη έκδοση της εφαρμογής, χρησιμοποιήθηκε η βιβλιοθήκη Knockout.js. Η Knockout ακολουθεί το πρότυπο MVVM, χρησιμοποιώντας δέσμευση δεδομένων (data binding) για να συνδέσει τη προβολή με το μοντέλο προβολής.

Για να δημιουργηθούν data bindings, μπορεί να προστεθεί ένα ειδικό χαρακτηριστικό data-binding για τα στοιχεία της HTML. Για παράδειγμα, η ακόλουθη σήμανση αντιστοιχεί το στοιχείο span σε μια ιδιότητα που ονομάζεται genre στο μοντέλο

προβολής. Κάθε φορά που η τιμή του genre αλλάζει, το Knockout ενημερώνει αυτόματα τον κώδικα της HTML:

```
<h1><span data-bind="text: genre"></span></h1>
```

Τα Bindings μπορούν επίσης να λειτουργήσουν και προς την άλλη κατεύθυνση. Για παράδειγμα, αν ο χρήστης εισάγει κείμενο σε ένα πλαίσιο κειμένου, η Knockout ενημερώνει την αντίστοιχη ιδιότητα στο μοντέλο προβολής.

7.7 Δημιουργία του μοντέλου προβολής (View Model)

Τα αντικείμενα observables είναι ο πυρήνας του συστήματος data-binding της Knockout. Observable είναι ένα αντικείμενο που αποθηκεύει μια τιμή και μπορεί να ειδοποιεί τους συνδρομητές όταν αλλάζει η τιμή.

Οι λειτουργίες επεξεργασίας προστίθενται αργότερα. Το μοντέλο προβολής περιέχει observables για τη λίστα των ταινιών, ένα αλφαριθμητικό σφάλματος και το τρέχον είδος.

Παρατηρείται ότι οι ταινίες είναι ένα observableArray. Όπως υποδηλώνει το όνομα, ένα observableArray ενεργεί ως ένας πίνακας που ειδοποιεί τους συνδρομητές όταν αλλάζει η σειρά των περιεχομένων.

7.8 Επεξεργασία εγγραφών

Το τελευταίο μέρος αυτής της εφαρμογής δίνει στον χρήστη την δυνατότητα να επεξεργαστεί τις εγγραφές στον πίνακα. Αυτό περιλαμβάνει πολλά κομμάτια λειτουργικότητας:

- Την εναλλαγή μεταξύ της λειτουργία προβολής (απλό κείμενο) και τη λειτουργία επεξεργασίας (controls εισόδου).
- Την υποβολή ενημερώσεων στον server.
- Το να επιτρέπει στον χρήστη να ακυρώσει μια επεξεργασία και να επιστρέψει στα αρχικά δεδομένα.

7.9 Δημιουργία Web Client με το Ember

Για λόγους σύγκρισης, σε αυτή την ενότητα παρουσιάζεται μια άλλη έκδοση της εφαρμογής με τη χρήση της βιβλιοθήκης Ember.js. Μια εφαρμογή Ember ξεκινάει με πίνακα δρομολόγησης, ο οποίος καθορίζει τον τρόπο όπου ο χρήστης θα πλοηγείται στην εφαρμογή.

```
window.App = Ember.Application.create();

App.Router.map(function () {

this.route('about');

this.resource('genres', function () {

this.route('movies', { path: '/:genre_name' });

});

});
```

Η πρώτη γραμμή του κώδικα δημιουργεί μια εφαρμογή Ember. Η κλήση για την Router.map δημιουργεί τρεις διαδρομές. Κάθε διαδρομή αντιστοιχεί σε ένα URI ή πρότυπο URI .

```
/#/about

/#/genres

/#/genres/genre_name
```

Για κάθε διαδρομή, μπορεί να δημιουργηθεί ένα πρότυπο HTML χρησιμοποιώντας το πρότυπο βιβλιοθήκης Handlebars.

Η Ember παρέχει ένα πρότυπο υψηλού επιπέδου για ολόκληρη την εφαρμογή. Αυτό το πρότυπο καθίσταται για κάθε διαδρομή.

7.9.1 Ελεγκτές και μοντέλα στην Ember

Στην Ember, κάθε δρομολογητής έχει ένα μοντέλο και έναν ελεγκτή. Το μοντέλο περιέχει τα δεδομένα του τομέα. Ο ελεγκτής (controller) λειτουργεί ως υποκατάστατο για το μοντέλο και αποθηκεύει όλα τα δεδομένα κατάστασης της εφαρμογής για την προβολή. (Αυτό δεν ταιριάζει ακριβώς με τον κλασικό ορισμό της MVC. Κατά κάποιο τρόπο, ο ελεγκτής είναι περισσότερο σαν ένα μοντέλο προβολής.)

Ένα μεγάλο πλεονέκτημα της Ember είναι ότι μπορεί να κάνει τα πράγματα με ελάχιστο κώδικα. Η Ember είναι επίσης ένα πολύ "δογματικό" πλαίσιο. Εάν ο

κώδικας δεν γραφεί με τον τρόπο της Ember, είναι πιθανό να υπάρξουν μερικά εμπόδια. Όταν επιλέγεται ένα πλαίσιο, θα πρέπει να εξετάζεται αν οι ρυθμίσεις των χαρακτηριστικών και ο συνολικός σχεδιασμός του πλαισίου ταιριάζει στις ανάγκες και στο στυλ κωδικοποίησης του προγραμματιστή.

Συμπέρασμα

Σε αυτό το κεφάλαιο, αναδείχτηκε πώς τα πλαίσια JavaScript κάνουν ευκολότερη τη δημιουργία των SPAs12. Στην πορεία, εισήχθηκαν ορισμένα κοινά χαρακτηριστικά αυτών των βιβλιοθηκών, συμπεριλαμβανομένου της δρομολόγησης .

[12] asp.net/single-page-application

Κεφάλαιο 8
Συμπεράσματα

Ποιο μοντέλο τελικά να χρησιμοποιήσω?

Μια κοινή ερώτηση είναι » ποιο μοντέλο είναι καλύτερο?». Σαφής απάντηση στην ερώτηση αυτή δεν υπάρχει. Κάθε μοντέλο έχει σχεδιαστεί για συγκεκριμένους σκοπούς και χρήσεις. Για παράδειγμα, οι Web Pages έχουν δημιουργηθεί με σκοπό να επιτρέπουν στους χρήστες που έχουν μικρή εμπειρία στο Web development να δημιουργήσουν βασικές σελίδες και διαδικτυακές εφαρμογές χρησιμοποιώντας την ASP.NET.

Οι διαδικτυακές εφαρμογές είναι μια ευρύτερη έννοια που περικλείει τις Web Forms και ASP.NET MVC και είναι σχεδιασμένες έτσι ώστε να είναι πλήρως λειτουργικές και επεκτάσιμες εφαρμογές (μπορούν επίσης να χρησιμοποιηθούν για να δημιουργηθουν Web Sites).

Άς δούμε συνοπτικά τα σημαντικότερα χαρακτηριστικά των μοντέλων.

- Web Pages (Ιδανικό για αρχάριους, small τέλειο για μικρές και μεσαίες εφαρμογές)

Οι Web Pages εξυπηρετούν έναν βασικό σκοπό. Είναι κατάλληλες να χρησιμοποιηθούν από προγραμματιστές με λίγη εμπειρία και λίγες γνώσεις, καθώς δίνει την δυνατότητα στους χρήστες να δημιουργήσουν πολύ απλές βασικές εφαρμογές και σελίδες. Το υλικό εκμάθησης τους είναι μικρό.

Η Microsoft συνεχίζει να δείχνει ότι ενδιαφέρεται για τις Web Pages καθώς συνεχίζει τις αναβαθμίσεις της WebMatrix platform τη οποία «φιλοξενεί» τις Web Pages.

Όσο το WebMatrix συνεχίζει να είναι μια βιώσιμη πλατφόρμα, τόσο θα υπάρχουν και θα εκσυγχρονίζονται οι Web Pages.

- Web Forms (υποστηρίζουν γρήγορη ανάπτυξη εφαρμογών, ιδανικές για εφαρμογές όλων των μεγεθών, υποστηρίζονται άψογα)

Όταν οι προγραμματιστές σκέφτονται την ASP.NET, την συσχετίζουν με τις Web Forms. Οι Web Forms μπορούν να θεωρηθούν μια πιο «επαγγελματικη» πλατφόρμα και ένα περιβάλλον πιο σύγχρονο από ότι οι Web Pages. Είναι το «κύριο εμπόρευμα» της πλατφόρμας .NET η οποία υπόσχεται γρήγορη ανάπτυξη και χαρακτηριστικά

που δουλέυουν με την μέθοδο drag-and-drop, με σκοπό την ανάπτυξη εφαρμογών που ποκίλλουν από μικρές μέχρι και εφαρμογές για επιχειρήσεις.

Οι Web Forms διαθέτουν μια τεράστια βιβλιοθήκη με στοιχεία ελέγχου που μπορεί εύκολα να ενσωματωθεί και παρέχουν μια εύκολη μέθοδο για την ανάπτυξη τόσο ιστοσελίδων όσο και ολοκληρωμένων διαδικτυακών εφαρμογών. Το υλικό εκμάθησης τους είναι μεσαίο.

Η δημοτικότητα των Web Forms έχει αρχίσει να φθίνει τελευταία χρόνια με την εμφάνιση της ASP.NET MVC, ωστόσο εξακολουθούν να είναι ένα πρότυπο για .NET Web Development που θα απασχολούν τα επόμενα χρόνια.

- ASP.NET MVC (Μεγάλο όγκο υλικού εκμάθησης, μεγαλύτερη ευελιξία και έλεγχος, ενσωματώνει απίστευτα καλά τις Javascript / client-side τεχνολογίες, ιδανικό για εφαρμογές όλων των μεγεθών)

Η ASP.NET MVC παρέχει έλεγχο υψηλού επιπέδου στις εφαρμογές σε σχέση με τις Web Forms και μια βλετιωμένη έκδοση των SoC “separation of concerns” (αρχές σχεδιασμού ενός προγράμματος σε διακριτά τμήματα, έτσι ώστε κάθε τμήμα ασχολείται με ξεχωριστή εργασία) μέσα στις εφαρμογές. Ωστόσο, η εμφάνιση της ASP.NET MVC δεν αντικαθιστά τις Web Forms αλλά παρέχει έναν εναλλακτικό τρόπο σχεδιασμού και υλοποίησης των εφαρμογών.

Το Model View Control είναι διαθέσιμο εδώ και αρκετό καιρό και έχει χρησιμοποιηθεί από διαφορετικά περιβάλλοντα και γλώσσες , όπως η *Ruby*. Το MVC έχει περισσότερο όγκο εκμάθησης σε σχέση με τις Web Forms, εξαιτίας της έλλειψης των “Controls”.

Το MVC αναπτύσσεται και διαδίδεται ραγδαία καθώς η δημοτικότητα του ολοένα και αυξάνεται.

Γίνεται εύκολα αντιληπτό ότι κάποιος μπορεί εύκολα να μπερδέψει αυτά τα μοντέλα και τις χρήσεις τους, γιατί φαινομενικά κάνουν το ίδιο πράγμα. Στην ουσία, είναι απλά διαφορετικά εργαλεία (κάποια μεγαλύτερα από κάποια άλλα) που τελικά υλοποιούν το ίδιο πράγμα. Οι ASP.NET Web Forms και το MVC έιναι δύο διαδικτυακά πλαίσια που αναπτύχθηκαν από την Microsoft και αποτελούν και τα 2 καλές επιλογές. Κάθε ένα από αυτά τα πλαίσια έχουν πλεονεκτήματα και μειονεκτήματα, τα οποία πρέπει να ληφθούν υπόψην όταν αναπτύσσεται μια διαδικτυακή εφαρμογή.

Μια διαδικτυακή εφαρμογή μπορεί να αναπτυχθεί με οποιαδήποτε τεχνολογία, μοντέλο. Μια εφαρμογή μπορεί να αναπτυχθεί ευκολότερα με την χρήση ενός μοντέλου και μια άλλη ευκολότερα με την χρήση ενός άλλου μοντέλου.

Συνοψίζοντας, να παραθέσουμε κάποια points :

- ASP.net web forms είναι ιδανικές για γρήγορη ανάπτυξη εφαρμογών.
- MVC είναι ιδιανικό για την βελτιστοποίηση των μηχανών αναζήτησης, καθώς μπορείς να ελέγχεις το URL και το παραγόμενο HTML σε μεγαλύτερο βαθμό.
- MVC γενικά παράγει μια πολύ λιτή σελίδα και είναι εύκολο να κρύψεις τμήματα της σελίδας.

Όπως βλέπετε, το ποιο μοντέλο θα χρησιμοποιηθεί εξαρτάται από τις απαιτήσεις μας και τις ικανότητες μας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Beck,K., *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 2000, ISBN 0-201-61641-6.

2. De Luca, J., Coad, P., & Lefebyre, E. (1999). *Java Modeling In Color With UML: Enterprise Components and Process*.

3. Schwaber, Ken; Beedle, Mike (2002). *Agile Software Development with Scrum*. Prentice Hall. ISBN 978-0-13-067634-4.

4. https://en.wikipedia.org/wiki/Client%E2%80%93server_model. [Accesed by 25/02/2017].

5. http://searchsoftwarequality.techtarget.com/definition/3-tier-application. [Accesed by 25/02/2017].

6. https://en.wikipedia.org/wiki/Multitier_architecture [Accessed by 2/22/2016].

7. http://webdesignergeeks.com/wp-content/uploads/2011/08/800px-Web_development_timeline.png [Accesed by 03/02/2017].

8. http://web-desktop-application.articles.r-tt.com/n [Accesed by 15/11/2017].

9. https://en.wikipedia.org/wiki/Web_application_development [Accesed by 5/03/2017].

10. http://www.comentum.com/guide-to-web-application-development.html [Accesed by 5/03/2017].

11. http://webdesignergeeks.com/web-designing/web-development-history-technologies/ [Accesed by 5/3/2017].

12. http://www.streetdirectory.com/travel_guide/114448/programming/desktop_applications_vs_web_applications.html [Accesed by 5/03/2017].

13. http://is.muni.cz/th/208139/fi_b/bc.pdf [Accesed by 13/12/2016].

14. https://www.asp.net/ [Accesed by 15/12/2017].

15. https://www.pluralsight.com/browse/software-development/web-development [Accesed by 27/03/2017].

16. http://www.sitepoint.com/successful-development/ [Accesed by 9/01/2017].

17. http://searchsoftwarequality.techtarget.com/definition/timebox [Accesed by 15/04/2017].

18. https://channel9.msdn.com/Forums/Coffeehouse/Questions-regarding-Web-pages-vs-Web-forms-vs-MVC [Accesed by 15/04/2017].

19. http://stackoverflow.com/questions/102558/biggest-advantage-to-using-asp-net-mvc-vs-web-forms [Accesed by 15/04/2017].

20. https://docs.microsoft.com/en-us/dotnet/articles/core/windows-prerequisites [Accesed by 25/04/2017].

21. https://docs.microsoft.com/en-us/aspnet/web-pages/overview/getting-started/aspnet-web-pages-razor-faq .

22. https://www.quora.com/Which-is-better-PHP-or-ASP-NET-and-why [Accessed by 03/05/2017].