Asynchronous Javascript And XML bez przeładowywania, w sposób asynchroniczny, dynamiczną interakcję.

1. [XMLHttpRequest](http://pl.wikipedia.org/wiki/XMLHttpRequest) - asynchroniczności w trakcie pobierania danych -> inne czynności -> dane jednocześnie z wielu miejsc
2. [JavaScript](http://pl.wikipedia.org/wiki/JavaScript) lub inny skryptowy
3. XML v Txt v Fragmenty HTML v JSON [ODPOWIEDZI od SERWERA]

Wady:

* treści strony poprzez język skryptowy ogranicza [dostęp do niej](http://pl.wikipedia.org/wiki/Dost%C4%99pno%C5%9B%C4%87_%28WWW%29) dla części użytkowników.
* Utrudnione jest automatyczne pobieranie stron (programy)
* indeksowanie przez serwisy wyszukujące może być utrudnione -> zadbać o linki wyświetlające całą stron v mapa witryny
* mozliwe problemy z pomiarami ogladalnosci
* brak obslugi przegladarkowego wstecz (nalezy zaimplementowac wlasne)
* przestaje nas dotyczyć podstawowy model/cykl życia strony internetowej

Zalety:

* szybsze dzialanie (reakcja strony i) -> lepszy user exp
* selektywne przesylanie - , zmniejszony transfer danych, lepszy czas ladowania

Language Integrated Query (.NET) warstwę abstrakcji miedzy app a danymi, pracujemy na obiektach stworzonych z np db, doc.xml, ADO.NET Datasets, kolekcje .net i inne
musza wspierać interfejs IEnumerable<T>. Skladnia linq jest bezposrednio oparta o kod i niezalezna od zrodla danych. Return kolekcja (IEnumerable<T>) Zapytanie LINQ może zwrócić: Zbiór obiektów, Pojedynczy obiekt, Podzbiór pól obiektów. Sa lambda wyrazenia
Wady: linq wykonuje sie dopiero przy uruchomieniu, trudnosci z dostrzeganiem bledu w fazie tworzenia

(Object Relational Mapping)) ADO .NET Entity Framework ,

ORM tworzy „wirtualną” bazę danych, która w konkretnych momentach jest synchronizowana z relacyjną bazą danych

Zalety

Większa produktywność (mniej kodu do napisania)

Warstwa abstrakcji nad relacyjnym modelem danych

Łatwiejsze zarządzanie operacjami CRUD na skomplikowanych strukturach danych

Łatwiejsze zarządzanie kodem aplikacji

EF może być stosowane w dowolnej technologii korzystającej z .NET

EF pozwala nam w łatwy sposób za pomocą takich metod jak SaveChanges, zapisywać stan danego modelu do bazy danych

Przyspieszenie pracy nad aplikacją. Od razu możemy pracować nad kształtem bazy danych.

Zmniejsza czas wytwarzania oprogramowania

Zmniejsza koszt oprogramowania

ORM dba o generowanie kodu SQL zgodnego z daną bazą danych. Zapewnia transparentny dostęp do różnego rodzaju relacyjnych baz danych. Takie rozwiązanie powinno ułatwić migrację danych między różnymi bazami.

Częściowe poradzenie sobie z problemem różnicy impedancji

Wady

Programiści co prawda nie musza znać modelu relacyjnego ale muszą stracić nieco czasu na nauce danego ORM’a

Ponieważ kod SQL jest generowany automatycznie, tracimy w pewnym sensie kontrolę nad tym co się dzieje.

ORM są niestety zazwyczaj nieco wolniejsze od rozwiązań bezpośrednio korzystających z bazy danych. W Internecie można znaleźć szereg różnego rodzaju statystyk, którą mowią np. o 10 do 30% wolniejszych zapytaniach.

Dzieję się tak ponieważ kod generowany nie jest optymalny

ORM nie zawsze radzi sobie ze skomplikowanymi zapytaniami

Ze względu na swoją „nieprzewidywalność” co do generowanego kodu, jest to rozwiązanie, które nie jest popularne w zastosowaniach wymagających dużej niezawodności.

ORM co prawda pozwala zaoszczędzić w początkowej fazie tworzenia oprogramowania, jednak generowany przez niego nieoptymalny kod SQL może się przyczynić do wzrostu kosztów administracji/utrzymania aplikacji

Niestety błędy związane z ORM’em są często trudne do wykrycia podczas debugowania i często ujawniają się później.

Często pojawiają się głosy mówiące o tym, że rozwiązania ORM niedługo okażą się przestarzałe.

Ponadto są osoby, które uważają takie rozwiązania za tzw. anti-pattern. Innymi słowy przykład niepoprawnego i błędogennego rozwiązania jakiegoś problemu.

MVC in webservices

Brak postbacku

Clean url -> korzysci SEO

Większa kontrola nad kodem HTML

Latwo testowac np unit testami

ASP.NET (Sesje, Cache’owanie i wiele innych)

latwo rozszerzalny

KIEDY MVC?

pełną kontrolą nad generowanym kodem

duzy zespol, testowanie, js

Windows Presentation Foundation (.NET) [korzysta z Model-View-ViewModel]

Architektura wyświetlania niezależna od rozdzielczości, 1 DPI=1/96 cala

eXtensible Application Markup Language(hierarchia i zagniezdzanie kontrolek, warstwa prezencji) hierarchia drzewa

Potężne wiązania danych redukujące ilość potrzebnego do napisania kodu

Różne dodatkowe funkcjonalności związane z tworzeniem layout’ów

Animacje 2D i 3D, wszelkie graficzne wodotryski (wektorowo, obiektowo) [Dependency Property]

Wspomagany sprzętowo (DirectX, CPU, GPU)

WebParts

* Personalizacja zawartości strony
* Personalizacja wyglądu strony
* Eksport i import kontrolek
* Tworzenie połączeń pomiędzy kontrolkami