Översikt

- Webbteknik från Microsoft.
- Klient/server.
- Designmönster.
- Utrullning.

Lektion 1: Webbtekniker från Microsoft

- Microsoft webbtekniker.
- ASP.NET.
- Klientsidan.
- Internet Information Server.

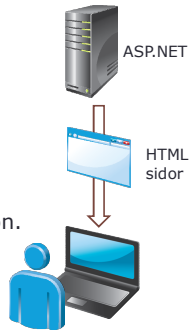
Microsofts webbtekniker

Utveckling	Värd	Exekvering	
		Server-Side	Client-Side
• WebMatrix (support slut nov. 2017). • Visual Studio • Visual Studio Code	• IIS • SQL Server • Windows Azure • SQL Database	• ASP.NET	• JavaScript • jQuery • AJAX

.eavec

ASP.NET

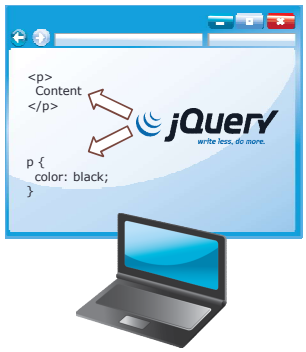
- Programmingsmodeller
  - Web Pages.
  - Web Forms.
  - MVC.
- ASP.NET API
  - Konfiguration.
  - Autentisering och auktorisation.
  - Mellanlagring.
- Kompilering av ASP.NET kod.



.eavec

Klientsida

- JavaScript.
- jQuery:
  - jQuery UI.
  - jQuery Mobile.
- AJAX.



.eavec

Internet Information Server

- IIS:
  - Funktioner.
  - Skalning.
  - Perimeter Networks (DMZ).
- IIS Express.
- Andra webbservrar.

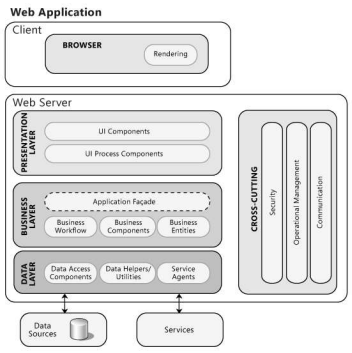
.easec

Lektion 2: Klient/server modell

- Klient/server modell.
- Att tänka på!
- Layout för sidan.
- Rendering.
- Navigering.
- Validering.
- Undantagshantering.
- Presentation layer.
- Data layer.
- Service layer.

.easec

Klient/server modell



.easec

Att tänka på!

- Dela upp din applikation logiskt.
- Förstå hur de olika komponenterna skall kommunicera.
- Använd mellanlagring.
  - Static caching.
  - Dynamic caching.
- Skicka inte känslig information oskyddat.
  - HTTPS.
  - Andra positive effekter.
- Designa din applikation att arbeta med ett konto med låga rättigheter.

.eosec

Layout för sidan

- Använd Cascading Style Sheets (CSS) för layout där det är möjligt.
- Table layout endast när information skall visas.
- Använd ett gemensamt utseende på sidorna.
- Använd AJAX server controls och klientbibliotek för AJAX, för att stödja olika webbläsare.

.eosec

Rendering

- Överväg att använda skript på klientsidan eller ASP.NET AJAX, för färre antal post back.
- För snabbare rendering använd data binding för controller.
- Skall olika språk användas för användaregränssnitt?

.eosec

Navigering

- Designa din navigerings-strategi på ett sådant sätt att den är separerad från logik för bearbetning.
- Använd navigeringsfunktioner i Master Page, så funktionen ser likadan ut för hela applikationen.
- Använd site map för att hjälpa användare att hitta sida på site.

Validering

- Designa en effektiv lösning för validering av indata är viktig för säkerheten och minska riskerna.
- Använd validering både på klientsidan och serversidan.

Undantagshantering

- Effektiv undantagshantering är viktigt för säkerheten och hur pass bra din applikation fungerar.
- Använd användbara felmeddelande som visar fel i applikationen utan att lämna ifrån känslig information.
- Försäkra dig om att du fångar undantag och rensar upp om ett undantag uppstår.

Presentation Layer

- Skiktet visar användare-gränssnitt och ger möjlighet till interaktion med användare.
- Skiktet består av server-side komponenter som renderar HTML och komponenter på klientsidan som exekverar skript och visar HTML.
- Använd AJAX för att exekvera logik på klient, oftast för att förbättra användarens upplevelse.

Data Layer

- Skiktet hanterar logik som är nödvänding för att få tillgång till databas.
- Gör det lättare att konfigurera och underhålla applikationen.
- Detaljer om databas göms för andra skikt i din applikation.

Service Layer

- Skiktet används om Web Service skall användas.
- Designa så att komponenter kan återanvändas.

Lektion 3: Designmönster

- Designmönster från Microsoft.
- Syfte med designmönster.
- Gang of Four.
- Creational Patterns.
- Structural Patterns.
- Behavioral Patterns.

.easec

Designmönster

- Microsoft har tagit fram ett antal mönster, patterns, som behandlar olika aspekter för utveckling.

.easec

Syfte med designmönster

- Designmönster har två syften:
  - gemensam plattform för utvecklare.
  - Best Practices.

.easec

Gang of Four

- Standardverk inom området mjukvaruutveckling är Design Patterns av Gamma, Helm, Johnson och Vlissides, 1994.
- I Design Patterns görs följande uppdelningar:
  - Creational Patterns (Skapande mönster).
  - Structural Patterns (Struktur mönster).
  - Behavioral Patterns (Beteende mönster).
- Varje designmönster har fyra beståndsdelar:
  - Ett namn.
  - En problemsituation.
  - En lösning.
  - En konsekvens.

.eaeec

Creational Patterns

- Designmönstren i denna klass (Abstract Factory, Builder, Factory Method, Prototype, Singleton) abstraherar instansieringsprocessen och hjälper till att konstruera system som inte är beroende av hur dess objekt skapas, komponeras och representeras.

.eaeec

Structural Patterns

- Designmönstren i denna klass (Adapter, Bridge, Composite, Decorator, Façade, Flyweight, Proxy) behandlar hur klasser och objekt genom arv och interface kan komponeras för att bilda större strukturer.

.eaeec

Behavioral Patterns

- Designmönstren i denna klass (Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor) behandlar algoritmer och fördelningen av ansvar mellan objekt.
- De beskriver inte bara mönster för objekt och klasser utan också mönster för interaktionen mellan dem.

.easec

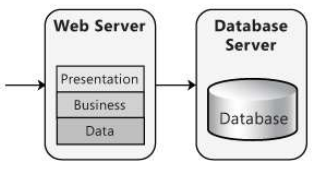
Lektion 4: Utrullning

- Non-distributed.
- Distributed.
- Lastbalansering.

.easec

Non-distributed

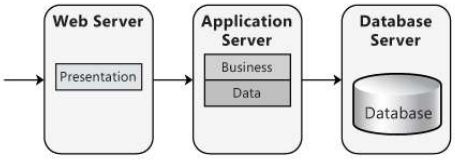
- All logik för de olika skikten är fysiskt placerade på samma webserver, förutom databas.
- Databas finns på separat server.



.easec

Distributed

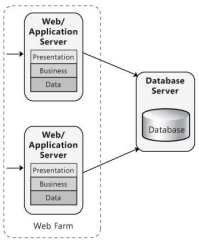
- Presentations- och business skikten är separerade.
- Oftast är business och data access skikten på samma server.



.eo/ee

Lastbalansering

- Om din applikation läggs på flera servrar kan lastbalansering mellan dessa användas, för att distribuera förfrågningar mellan dessa.
- Viktigt att hantera state.



.eo/ee

Repetitionsfrågor

.eo/ee