React

Uvod:

Svijet komponenata – angular, jquery, bootsrap, ember (slika frameworka)

Specificnosti reacta – jsx, virtual dom, isomorphic rendering, jednosmjerni tok podataka

Html treba biti projekcija stanja aplikacije(podataka) – prije je stanje bilo u DOM-u

Js i HTML su u istom file-u

Jesnosmjerni tok podataka – nema two way bindinga

(slika webova koje koriste reacta)

Koristenje tehnologija:

Nodejs – to run server side js

* Npm package manager – (nuget npr.)
* commonJS pattern – enkapsulacija js koda u module (include, import i sl.)

Browserify – expose packages to the browser – bundla module u jedan file

React – library za komponente

React router

Flux

Gulp

DEMO1

React koncepti:

* brz, kompozicija komponenti, lako se koristi u postojecim aplikacijama, isomorphic friendly, jednostavan, dobro testiran (na facebooku)
* MVC – react je V ( i C)
* Rizici two-way bindinga – nepredvidljivo (kaskadiranje promjena), tesko debugiranje
* JSX – „html“ u js-u, nije html ( prevodi se u js) – sluzi za definiranje markupa
* Separation of concerns?? – toga vise i nema na webu. Vecina koristi js u html-u
  + Html i js uvijek moraju biti cvrsto i dobro siknronizirani – tesko jer nema definiranog interface-a
  + U jsx-u compiler kaze gdje je greska, u html-u samo ne radi – prednost jsx-a
* Virtual DOM – usporedba proslog i buduceg stanja – update samo potrebnog dijela DOM-a
  + Updateanje DOMa je skupa operacija
  + Npr brisanje reda tablice – iscrtavanje nove tablice vs izbacijvanje jednog reda

DEMO2 – jednostavna komponenta

DEMO3 – krearanje komponent

* Komponenta
* Custom routing
* Centralizirani layout

Reac lifecyle – props kao html atributi – slanje podataka child kontrolama - nepromjenjive

and state – nepromjenjive – top level komponente drze podatke u njima

* Lifecyle funkcije
  + Component will mount – prije renderiranja – dohvat inicijalnog stanja
  + Component did mount – poslije renderiranja – ajax pozivi npr
  + componentWIllreceiveProps – kod primanja novik props-a(ne poziva se kod inicijalnog renderiranja) – postavljanje stanja prije novog renderiranja
  + shouldComponentUpdate – prije novog renderiranje – sluzi za sprijecavanje nepotrebnih renderiranja
  + componentWillUpdate - tocno prije renderiranja(ne kod inicjalnog rendera) – priprema za update doma
  + componentdidUpdate – odmah nakon novog iscrtavanja
  + componentWillUnmount – netom prije micanja kontrole iz doma

kljucevi za dynamicne child kontrole

* potrebno koristit nekakav uniqeId kako bi parent znao razliokovati child kontrole

DEMO4

React composition:

ControlerView - top level kontrola koja se bavi upravljanjem podacima i prosljedjuje ju child kontrolama (preko propsa)

Prop validation – definiranje tipa propertya kakve kontrola ocekuje – samo u development modeu (production mode se izvrsava na produkciji)

DEMO5

React Router

Routes

Links

DEMO6

React forms :

DEMO7

Flux:

* Pattern a ne framework – mnogo implementacija jednosmjernog toka podataka
* Redux trenutno najpopularniji
* Mi cemo vidjet facebook flux
* (slika mvvm vs flux)
* Viewmodel - view
* Action– dispatcher – store – view
* Actions:
  + Enkapsulira evente
  + Uzrokuju je useri interakcijom preko UI
  + Prosljedjuju se dispatheru
  + Type + data
* Dispatcher
  + Singleton
  + Centralni sustav callbacke
  + Obavjesta store(one koje to zanima) o akcijama
* Stores
  + Cuvaju stanje(podatke) aplikacije
  + Moze ih biti vise
  + Imaju callbacke registrirane kod dispatchera
  + Koriste eventEmitter za emitiranje evenata viewima
  + „hej dispatcher, ako se nesto dogodi obavjesti me“
  + Top level componanta komunicira sa storama i drza podatke u state-u

DEMO8