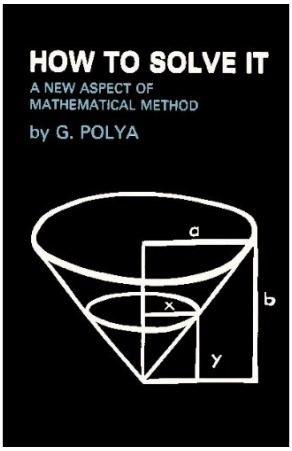
Programming -> Problem Solving



Architecture Briefings

@stuarthalloway

Choice -> Power

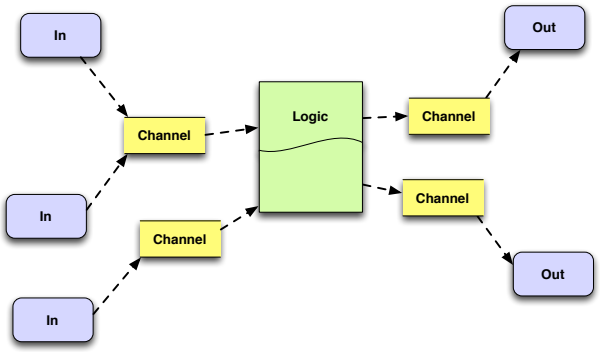
IT -> Competitive Advantage

cognitect

http://clojure.org/rationale

https://github.com/clojure/clojurescript/wiki/Rationale

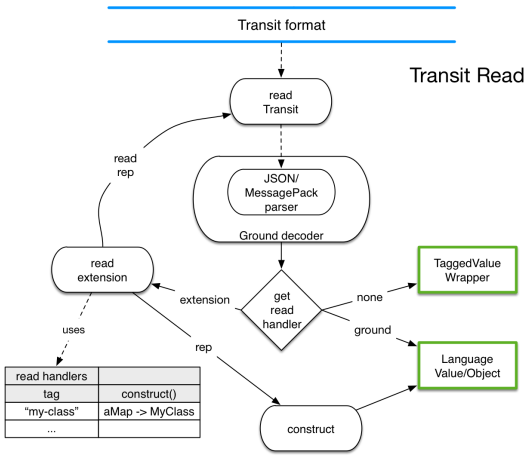
http://clojure.com/blog/2013/06/28/clojure-core-async-channels.html



http://www.datomic.com/rationale.html

Datomic

https://github.com/cognitect/transit-format#rationale



NIH bias? Hardly!

Clojure brings [*] to the JVM

ClojureScript brings Clojure to the Browser

core.async brings CSP to everybody

Transit brings good semantics to JSON

How Do You Make Technology Decisions?

How Not To Do It

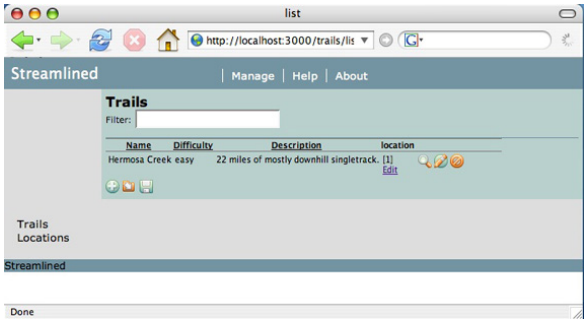
~~developer convenience~~

~~familiarity~~

~~widespread adoption~~

~~hands-on experience~~

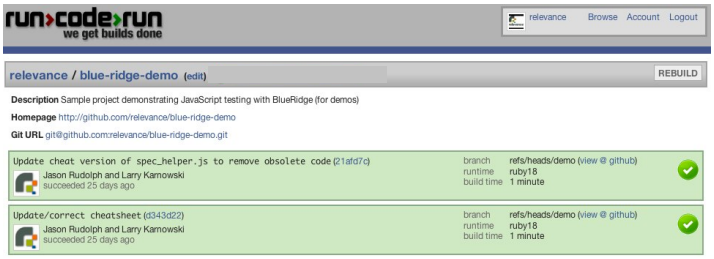
coding as coding



http://www.ibm.com/developerworks/library/j-cb09056/

coding as thinking

(shh... we tdd'ed a web appinto a queue)



Use Your Nose?

http://www.paulgraham.com/javacover.html

http://java.dzone.com/articles/how-hackers-choose-tools-0

http://en.wikipedia.org/wiki/Paul_Graham_(computer_programmer)

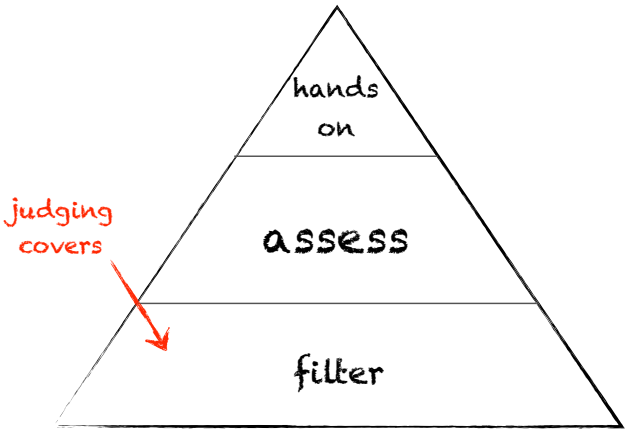
Judging Covers

leadership matters

community matters

safety vs. power

rationales matter



Learning Initiative

internal use only

regular time, bacon

everybody presents

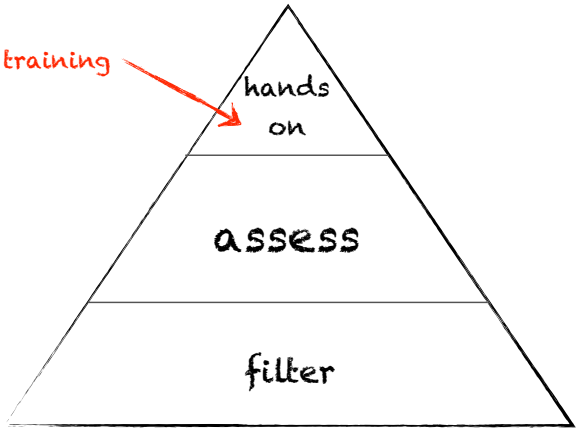
reviewed in advance

distributed & recorded

collect feedback

nothing fancy

the "skeletal" parts only



training: overview

1 hour preso, 30 min exercise

focus on 1 topic

how-to

low-friction setup

be ready for broader questions

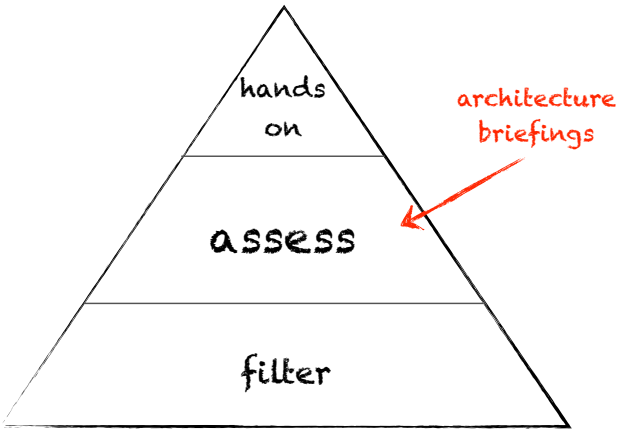
training: template

when to use (recaps arch briefing)

in use

summary

exercise prep



architectural briefings

1 researcher, N people up to speed

why X?

what do you need to know to answer "why"?

having chosen X, what do you need to know to apply it?

attendee guide

take notes

question everything

what is missing?

why?

compare with your own experience

do you agree?

you are up next

briefing overview

45 minutes max

high-level briefing on *specific* tech

no how-to

go beyond the official docs

be prepared for questions

outline (1)

what is it

what problem does it solve

fundamental semantics

- what is it about
- what are the primary operations
- is it simple?

outline (2)

key indicators for/against use

fundamental tradeoffs

alternatives

- any worthy of a briefing?

cost

- even free/OSS has costs

outline (3)

environment

docs

API quality

deps

community

longevity

experience reports

outline (4)

- throughput
- latency
- elasticity
- scalability
- deployment
- monitoring
- security
- failure modes

Sample Briefing

Russ Olsen - To the Moon! - YouTube
www.youtube.com/watch?v=4Sso4HtvJsw

https://www.youtube.com/watch?v=4Sso4HtvJsw

Vaadin*

Architectural Briefing

Cognitect Learning Initiative
Russ Olsen

*Rhymes with nothing at all

Choosing To Use

What Is It?

- Vaadin is a Java web application framework
  - Includes server side Java engine
  - Includes browser side engine
  - Includes a library of UI components
  - Includes tools for building UI components in Java*

*Not addressed here

What Problems Does Vaadin Try to Solve?

- Problem 1: Building web applications requires knowing Clojure, JavaScript, HTML, CSS
- Problem 2: Building web applications requires coming up with a client/server communication scheme
- Problem 3: Building web applications require writing a lot of tedious code

What Problems Does Vaadin Strive to Solve?

Make the creation and maintenance of high quality web-based user interfaces without having to deal directly with browser technologies like HTML or JavaScript.

Fundamental Semantics
What Is Vaadin About?

Vaadin takes building web UIs back to the world of Swing or SWT:

- You build your UI with server side code (Java or Clojure)
- Vaadin takes care of the HTML, the JavaScript and the CSS
- Vaadin takes care of the browser/server communication

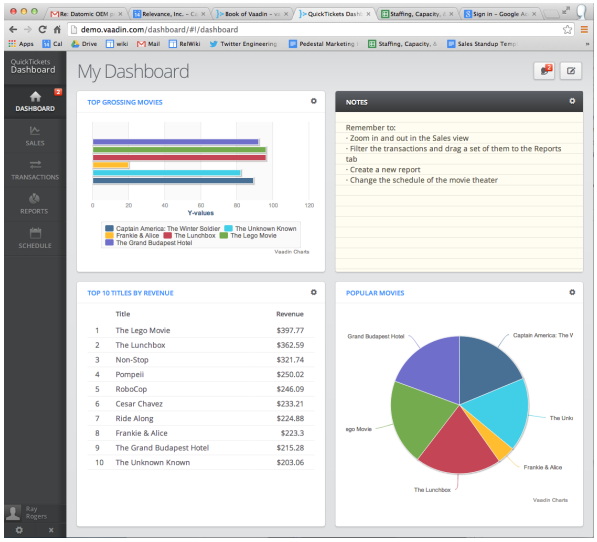
Peek at the Code

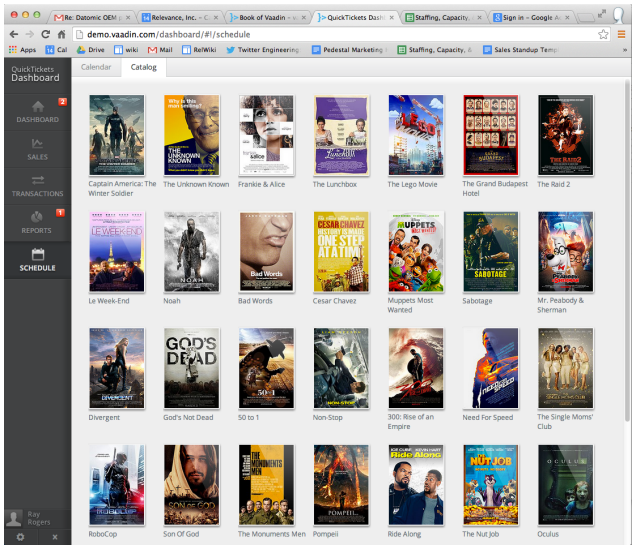
```
(ns cljvaadin7.MyApplicationUI
  (import [com.vaadin.ui CssLayout Label Button TextArea])
  (:gen-class
    :extends com.vaadin.ui.UI))

(defn -init [main-ui request]
  (let [layout (new CssLayout)]
    (doto layout
      (.addComponent (Label. "hello world from clojure"))
      (.addComponent (Button. "Push me"))
      (.addComponent (TextArea.)))
    (.setContent main-ui layout)))
```

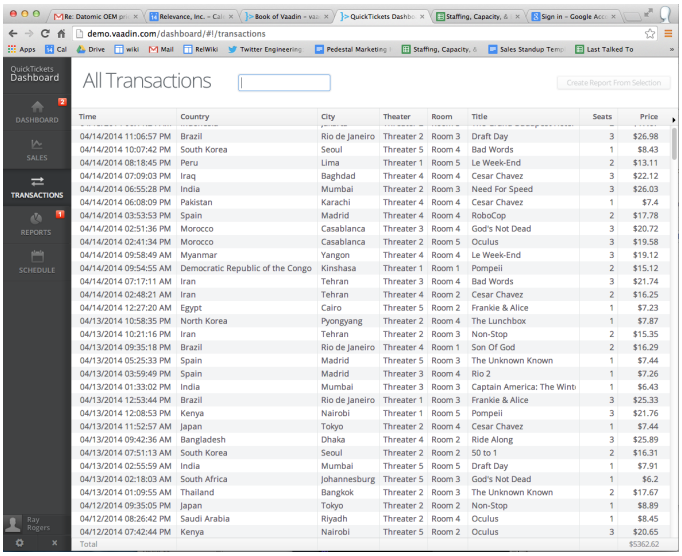
Fundamental Semantics
What Is Vaadin About?

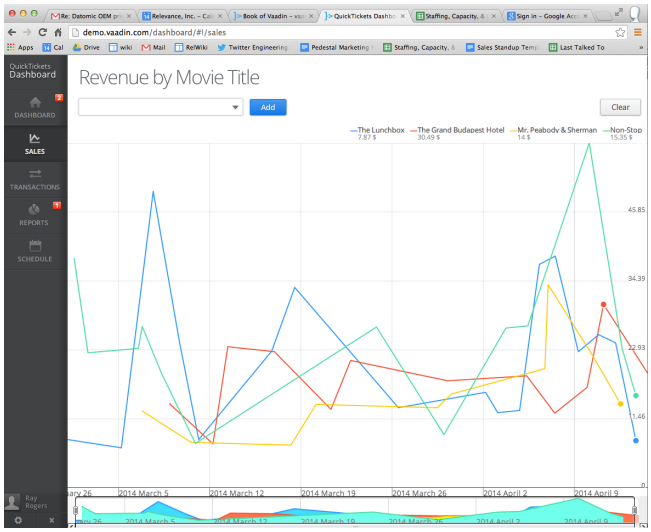
Vaadin lets you forget that some of your application lives on the server and some on the browser.

Mostly









Fundamental Semantics
What Are the Primary Operations?

- You program a tree of components on the server side that represents your UI
  - You add listeners to buttons and data components
  - Optionally you can style your components with SASS
  - Optionally you can push data out to the client
- You then build a WAR file and deploy it to a servlet container such as jetty or tomcat

Fundamental Semantics
Is It Simple?

Fundamental Semantics
Is It Simple?

Our mission is to make building amazing web applications **easy**.*

Fundamental Semantics
Is It Simple?

No

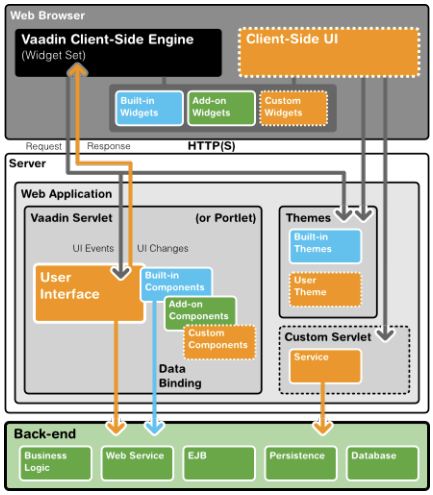
*http://vaadin.com/company

Fundamental Semantics
Is It Simple?

- Vaadin has a server side library, a set of client/ server communication conventions, a collection of components, each with a client and server side
- Vaadin's "It's all on the server" abstraction leaks a bit
- Useful nevertheless

Architectural Overview
Components

- Client side engine
- Servlet and Portlet
- Components: TextField, Label, MenuBar, etc
- Layouts: VerticalLayout, TabSheet, CSSLayout
- Themes
- Bring your own server (tomcat, jetty, Glassfish)



Architectural Overview:
Key Features

- Server side UI programming
- Browser side UI programming*
- Many components available:
  - Charts
  - Lazy loading
  - Specialized components for mobile support
- Drag & Drop support
- Server push (WebSocket by default)
- Customizable themes via SASS
- Deep link support
- Back button support

* Not covered here

Indicators *For* Use

- Web app running on the JVM
- Building an app with a significant UI
  - Data input, forms, charts
  - Especially if the UI is "screen-y": Think tracker
  - There are existing components that fit your app
- Modest number of users

Indicators *Against* Use

- Not on the JVM
- No UI or minmal UI
- Need very fine grained control of the UI
  - Need to match some an existing UI exactly
  - There is no Vaadin component for that
  - Very Page-y interface
- Need fine grain control of Browser/Server communication
  - Existing server API?

Indicators *Against* Use (cont)

- Possibly very high performance
  - Vaadin is reported to be chatty
    - Events get sent to server
    - Deltas get sent back to browser
  - Events must find their way back to the server side model

Fundamental Tradeoffs

- You get:
  - Highly simplified model for building user interfaces
  - Relieve from having to worry about every mouse click
  - Relief from having to worry about every single pixel
  - Relief from having to worry about browser / server communications
- You give up:
  - Pixel by pixel control of UI look
  - Full control of the user experience
  - Control of the browser / server communications
  - Stateless server

Alternatives

- Java frameworks with very similar programming model
  - Echo: http://echo.nextapp.com/site/echo3
  - Wicket: http://wicket.apache.org
- These might be worth a second look

Alternatives (cont)

- Java frameworks with somewhat similar programming model
  - ZK: http://www.zkoss.org
  - ICEfaces: http://www.icesoft.org/java/home.jsf
- Both ZK and ICEfaces require you to write some markup.

Alternatives (cont)

- Assemble your own app with:
  - Om
  - Backbone
  - Ring
  - etc

Application Characteristics
Environment

- You deploy a Vaadin app as either a servlet, or as a portlet:
  - Java >= 6
  - Servlet API >= 2.4, Portlet 2.0
  - Compatible with Google App engine, Tomcat, JBoss, Jetty, etc.
- Recent versions of major browsers supported
  - Chrome 23+, IE 8+. Safari 6+
- For more see https://vaadin.com/features

Application Characteristics
Documentation

- Copious, well written and apparently complete.
- Main reference is *The Book of Vaadin*
  - About 600 pages of clear explanation, complete with examples.
  - https://vaadin.com/download/book-of-vaadin/vaadin-7/pdf/book-of-vaadin.pdf
- API docs:
  - https://vaadin.com/api/
- Keep in mind, 99.9% of this documentation is about using Vaadin in Java not Clojure.

Application Characteristics
API Quality

- Good
  - Mostly avoids the “maze of twisty Java classes” syndrome
  - Appears to have profited by looking at the Swing, SWT Apis and discarding the cruft
- Usable directly from Clojure w/o wrapping
- Callbacks work well with core.async

Application Characteristics
Dependencies

- A minimal Vaadin application depends on the following Maven artifacts, all with group id com.vaadin
  - vaadin-server
  - vaadin-client-compiled
  - vaadin-client-compiler
  - vaadin-client
  - vaadin-push
  - vaadin-themes

Application Characteristics
Dependencies (cont)

- com.google.appengine/appengine-api-1.0-sdk
- com.liferay.portal/portal-service
- javax.portlet/portlet-api
- javax.servlet/servlet-api
- javax.validation/validation-api
- org.jsoup/jsoup
- org.w3c.css/sac
- com.vaadin.external.atmosphere/atmosphere-runtime

Application Characteristics
Dependencies (cont)

- You also need servlet container such as Jetty or Tomcat or Glassfish
- Alternatively you can deploy your Vaadin application as a portlet in a portal such as LifeRay.

Community
Longevity

- Been around in one for or other since 2002
- Supported by a commercial company
- @vaadin has about 3,300 twitter followers
- Reasonably active forum (10’s of posts per day)
- Meetups in CA, several in Europe

Community
Experience Reports

- Bobby et. el. on my-datomic dashboard
  - git@github.com:Datomic/my-datomic-dashboard.git
- Tim et. el. on datomic-console
  - git@github.com:Datomic/my-datomic-dashboard.git

Community
Experience Reports

- Both Tim & Bobby had good experiences
  - Works
  - Delivered on the good-UI-for-less-work promise
  - Integrates well with Clojure
    - Component API is reasonable
    - Plays well with core.async

Operational Characteristics

- Throughput
- Latency
- ~~Elasticity~~
- Scalability
- Deployment
- Monitoring
- Security
- Failure modes

Operational Characteristics
Throughput

- A concern since the Vaadin is reported to be chatty
- Artificial benchmark simulating movie ticket purchasing measured 2748 purchases / min @ 1% reject rate on EC2 large instance using RDS
- Source: https://vaadin.com/blog/-/blogs/vaadin-scalability-study-quicktickets
- Please add a gain of salt

Operational Characteristics
Latency

- Same benchmark measured
  - 92ms response @ 5,000 users
  - 177 ms response @ 11,000 users
  - 251 ms response @ 13,000 users
- Should also be taken with a grain of salt

Operational Characteristics
Scalability

- Scaling number of users:
  - Per session memory requirements may be higher than a hand coded application
  - Requests need to find their way back to the server side object model
  - Sticky sessions or session replication
- Scaling complexity of UI and underlying data:
  - Some concern about the amount of markup generated for even simple interfaces
  - Lazy components provide support for viewing large datasets

Operational Characteristics
Deployment

- Vaadin applications deploy as Java web applications (i.e. a WAR file) into an server like jetty or tomcat
- Can also be deployed into a JSR 286 portal

Operational Characteristics
Monitoring

Since Vaadin deploys as a standard Java web application, you can use any standard JVM based tool for monitoring

Operational Characteristics
Security

- Application data mainly lives on the sever, not the browser
- Data validation lives on the server
- Vaadin includes built-in protection of XSS and Request Forgery
- Authentication & Authorization
  - Not included
  - You are free to use whatever makes sense: JAAS, Spring Security, Apache Shiro, etc.

Operational Characteristics
Failure Modes

- Browser side components loses contact with the server side model
  - User session gets reset
  - User probably loses any unsaved data

Operational Characteristics
Failure Modes

- Browser side components get out of sync with server side components
  - User sees an apparently functional but semi to fully broken interface
  - Happened to me (once) on an unreliable network
  - Scattering of reports on the internet

What Does It Cost?

- Framework & Basic components are free/open source
  - Apache License 2.0
  - Generally well regarded
- Support is available $4,000 - 16,000 - wadda got?
- Additional components available as open source
  - Quality varies

What Does It Cost? (cont)

- Chart components: $500 / dev
- Touchkit (mobile) components $600 / dev
- Selenium based test tool $900 / dev

What Does It Cost? (cont)

Call Now!

Or buy all 3 for the low, low price of $500 / dev month!

Call Now!

Questions?

In Use

What Are the Critical Decisions?

- How do I ensure that requests make it back to the server side model?
- How do I bind my components to their data?
- How do I control UI layout?
- Do I use Vaadin for my entire UI?
- Do I use server side push?

How Do I Ensure that Requests Make It Back to the Server Side Model?

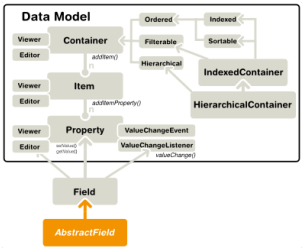
- Single server with failover?
  - Pro: Simple to set up
  - Con: One server
  - Con: User loses data on server failure
- Sticky sessions?
  - Pro: Multiple servers
  - Con: User loses data on server failure

How Do I Ensure that Requests Make It Back to the Server Side Model? (cont)

- Session replication
  - Pro: Provides continuous operation on server failure
  - Con: Reported to slow down the application, use more memory per user
  - Con: Not well covered in Vaadin documentation
  - Con?: Lots of questions “Why doesn’t this work?”

How Do I Bind My Components to their Data?

- Vaadin comes with a built-in data model
- Con: A little bit of Java class Hell
- Pro: May be worth it for the magic of SQLContainer

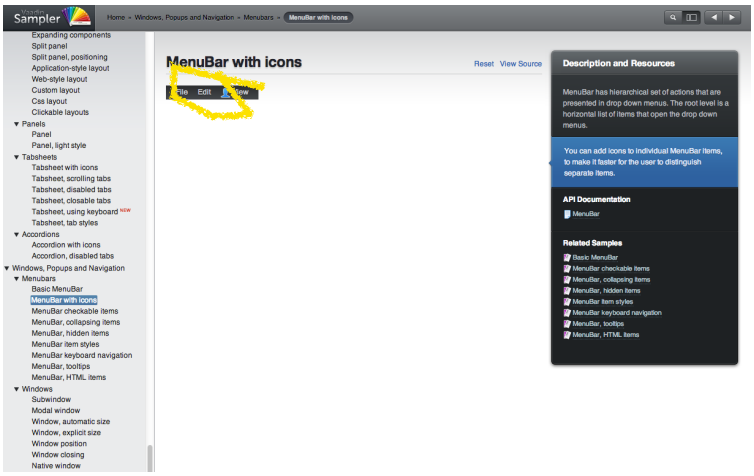


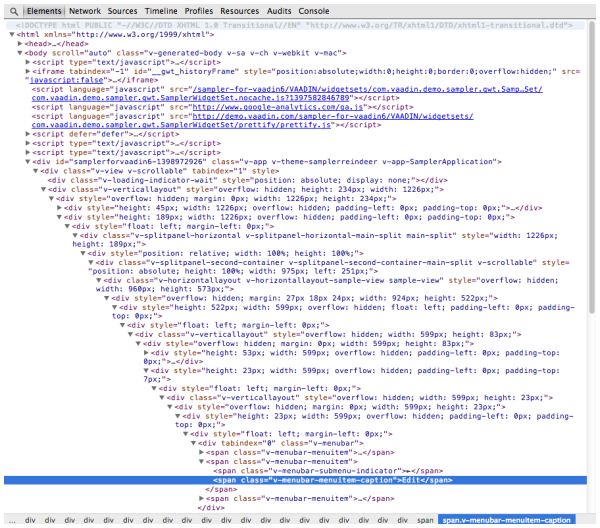
How Do I Bind My Components to the Data? (cont)

- Alternatively you could simply construct callbacks for component events and bind to the data that way
  - Pro: Simple
  - Con: …but gets you into callback Hell
- Or you could use core.async
  - Pro: Just about as simple
  - Pro: … and no callback Hell

How Do I Control UI Layout?

- Relative layout components
  - VerticalLayout, HorizontalLayout, GridLayout
  - Pro: Easy to reason about
  - Con: Less control
  - Con: You are in a maze of twisty divs, all of them different (see next 2 slides)
- CSSLayout
  - Pro: You have CSS control
  - Pro: Fewer divs
  - Con: You gotta write the CSS





Do I Use Vaadin for My Whole UI?

- Yes
  - Pro: The code is simpler, easier to reason about
  - Con: You are limited to what Vaadin can do

Do I Use Vaadin for My Whole UI? (cont)

- No
  - Next question: Do you have HTML with Vaadin inside or Vaadin with HTML inside
  - Pro: You can do things that Vaadin can't
  - Con: Need to deal with layout, JS interaction issues
  - Con: Need to match the look of raw HTML & Vaadin app

Do I Use Push?

- Pro: Push gives you an asynchronously updating interface
- Con: Pushing involves locking the server side components
- Con: Push documentation is scanty
- Con: Feature does not seem fully baked

Questions?

Summary

Questions

- How do people handle an out-of-sync client?
  - Not clear
- How does session replication work?
  - Not clear
- What does the Vaadin protocol look like?
  - https://vaadin.com/book/-/page/gwt.shared-state.html

Good Introductory Paper

- Introduction to Book of Vaadin
  - https://vaadin.com/book/vaadin7/-/page/intro.html
- Good article about using Vaadin in Clojure
  - https://vaadin.com/wiki/-/wiki/Main/Using+Vaadin+and+TouchKit+with+Clojure
- Clj-reindeer
  - Vaadin to Clojure Wrapper library
  - Not sure how good the library is, but good source of example code
  - https://github.com/feldi/clj-reindeer

Essential Reference Books

- Book of Vaadin
  - https://vaadin.com/book
- Vaadin 7 Cookbook by Holan and Kvasnovský
  - Available on Amazon
  - Good ratings, preview is very limited

What's Next?

- Vaadin (the company) seems focused on incremental improvements
  - Version 7.2
    - Improved layouts, push support for more app servers, performance
    - Expected May 2014
  - Version 7.4
    - Better grid component
    - Expected mid summer
  - Version 7.5
    - And so on

New Alternatives?

Didn't find any

Use It?

- If you are building an app with a UI with significant amount of data input and/or charts
  - And you don't need pixel by pixel, click by click control
    - And the number of users is modest
      - Then Vaadin is a no brainer
- If all of the answers are not "yes" then you will have to use your brain.

Final Thoughts

Vaaden is not how we have traditionally built applications.

Tradition is not a real data point.

concluding thoughts

Questions?

assessment > hands on learning

many assessment methods out there

- match your culture and objectives

structure matters