QUICK RECAP

A reminder of what we have done during Lecture 01

LAST TIME…

- Desktop applications
  - What are they?
- Web applications
  - What are they?
  - How they differ from desktop apps?
  - What is the common architecture of web apps?
  - What technology can be used?
- Vaadin
  - What is it?
- Shoutbox app

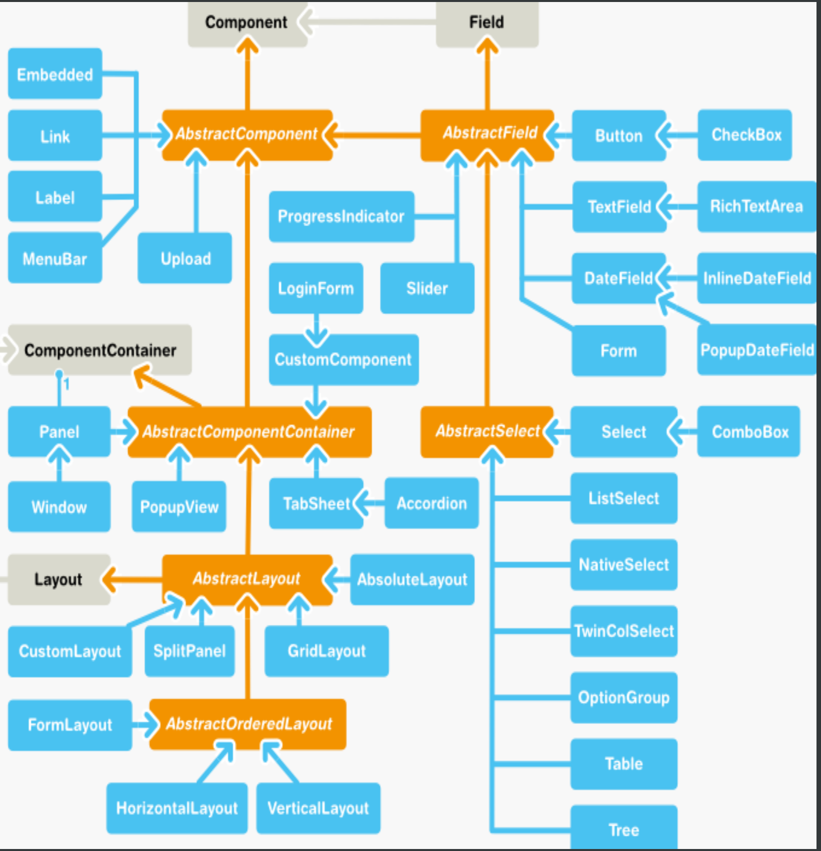
VAADIN FRAMEWORK: COMPONENTS

Development of Modern Web Applications (with Vaadin)

Lecture 02

USEFUL RESOURCES

- https://demo.vaadin.com/sampler
  - Live and up-to-date
  - With source code samples
- https://vaadin.com/refcard
  - Cheat sheet
- https://vaadin.com/book
  - The Book of Vaadin
- https://vaadin.com/api
  - JavaDocs



COMPONENT HIERARCHY

Legend

Gray	interface
Orange	abstract class
Blue	concrete class

Two subhierarchies

Components	(including containers)
Fields	(including selects)

Five groups of components
- "Bare" components
- Containers
- Layouts
- Fields
- Selects

COMPONENT

- Top of the class hierarchy
  - Paired with `AbstractComponent`
- Parent-child relationship at runtime
- Almost everything is a `Component`
  - Thus everything shares some common properties

COMMON PROPERTIES

Everything is a component

● ● ● ●

PROPERTY: CAPTION

- An explanatory text accompanying a user interface component
  - Displayed near
  - Shortly describes purpose or role
  - Plain text only
    - HTML escaped
- Handling varies
  - Default is to leave it for layouts
  - Some components display the layout by themselves

Property: Caption

Server side

```
foo.setCaption("bar");

String bar =
     foo.getCaption();

// nothing fancy here
```

CSS Rules

- .v-caption
  - Caption element
- .v-captiontext
  - Caption itself
- .v-caption-clearelem
  - Clears floating elements

PROPERTY: ICON

- An explanatory graphic accompanying a user interface component
  - Displayed near
  - Depicts role
    - Accessibility is important
- Handling varies
  - Some display themselves
  - Some rely on parents

Property: Icon

Server side

```
// sets icon based on
// current theme

foo.setIcon(
 new ThemeResource(
  "icons/user.png"
));

// there are several
// types of resources

Resource icon = foo.getIcon();
```

CSS Rules

- .v-icon
  - May be inside .v-caption

PROPERTY: DESCRIPTION

- More elaborated text about the component
  - Detailed information, e.g.
    - Component role
    - Allowed values
  - Rich text
    - (X)HTML allowed
- Usually a tooltip

Property: Description

Server side

```
foo.setDescription(
  "<h2>Hello, world!</h2>");

String bar =
        foo.getDescription();

// notice that it
// can be anything
// e.g. <img>
```

CSS Rules

- No CSS rules
- Painted by the client

Property: Locale

Overview

- Inheritable
  - Parent
  - Application
  - System
- Not available in the constructor
  - No straightforward i18n
- No CSS rules

Server side

```
// language ISO 639-1
// country ISO 3166-1-
//            -alpha-2

foo.setLocale(
  new Locale(
    "pl", "PL"
 ));

Locale lc = foo.getLocale();
```

Flag: Enabled

Overview

- Concerns user interaction
- Affects all contained components
  - Disabled
    - Visible
    - Accepts no user input
  - Enabled
    - Default state

Code

- Java

```
foo.setEnabled(boolean);
    boolean bar =
        foo.isEnabled();
```

- CSS rule
  - .v-disabled
    - In addition to normal CSS
    - No rule for enabled components

Flag: Visible

Overview

- Concerns component visibility
- Affects all contained components
  - Invisible
    - As if never existed
    - Still available in Java code
  - Visible
    - Default state

Code

- Java

```
foo.setVisible(boolean);
 boolean bar =
     foo.isVisible();
```

- No CSS
  - Empty element
  - Inline style
    - `display: none;`

Flag: Read-only

Overview

- Concerns user interaction
- Does <u>not</u> apply to contained components
  - Read-only
    - Value modifications not communicated by the browser
    - Server does not accept any changes
      - `Property.ReadOnlyException`
  - Non read-only
    - Default state

Server side

```
foo.setReadOnly(boolean);
boolean bar =
          foo.isReadOnly();


// css: .v-readonly
// in addition to
// normal CSS
// no CSS for default
```

PROPERTY: STYLE NAMES

- Custom CSS style class names
  - Name your things!
  - Theming done in app's CSS
  - Style helpers available
    - And much recommended
- Rendered in two forms
  - .v-{component}-{style_name}
  - .{style_name}
- Should not cause conflicts
- Must be valid CSS class name

```
Label foo =
  new Label("foo");

// sets style name or names
foo.setStyleName(
 "style1 style2 style3"
);

// add and remove
// one-by-one
foo.addStyleName("style1");
foo.addStyleName("style2");
foo.removeStyleName("bar");

// space-separated string
String bar =
         foo.getStyleName();
```

STYLE NAMES IN CSS

Per component:

    .v-label-style1
    .v-label-style2
    .v-button-style1

Per style:

    .style1
    .style2

vaadin}>

PROPERTY: WIDTH AND HEIGHT

- Defined in interface `Sizeable`
  - Which is extended by `Component`
    - It probably makes some sense for any component
- `setSizeUndefined()`
  - The component will take as little space as possible
  - Recommended approach is to set size in CSS
- `setSizeFull()`
  - Recommended approach when taking all the space available
- Default behaviour varies
  - Most components are undefined
  - Layouts may have 100% one way or another

PROPERTY: WIDTH AND HEIGHT

There are no CSS rules – if specified, the size is rendered as an inline style (cannot be overriden)

```
Button foo =
  new Button("foo");

// sets width with CSS string
foo.setWidth("100px");

// set height another way
foo.setHeight(
      50, Sizeable.Unit.PERCENTAGE);

// sets height undefined
foo.setHeightUndefined();

// shortcuts for setting both dimensions
foo.setSizeUndefined();
foo.setSizeFull();

// getters
float width = foo.getWidth();
Sizeable.Unit widthUnit = foo.getWidthUnit();

// similar for height, of course
```

PROPERTY: FOCUSABLE AND TABINDEX

- Defined in `Component.Focusable`
    - Not every component can be focused
    - Each `Field` is focusable
- No way to find currently focused component
    - Some (but not all) fields broadcast
        - `FocusEvent` when receiving focus
        - `BlurEvent` when losing focus
- Tab index manages keyboard navigation
    - Yes, people use that
    - Negative index makes component not available for tabbing
    - Default tab order follows run-time hierarchy

Property: Focusable and TabIndex

Server side

```
TextField foo =
  new TextField("foo");

// server-side focusing
foo.focus();

// tab-order
foo.setTabIndex(1);
int index =
        foo.getTabIndex();
```

CSS Rules

- .v-{component}-focus
  - In addition to normal CSS
  - No rule for no focus

vaadin}>

SIMPLE COMPONENTS

A short overview

com.vaadin.ui.Label

ContentMode.TEXT

- Default mode
- Plain text
- < > & are escaped

ContentMode.PREFORMATTED

- Monospaced font
- Can contain \n \t
- < > & are escaped

ContentMode.HTML

- Rendered as <div>
- Should be XHTML 1.1 Strict
- Can lead to cross-site scripting

Random trivia

It has a caption, which is something different than its value

COM.VAADIN.UI.LINK

- Clickable link to an external resource
  - Target resource can be changed
- Can have an icon
- Opens in current window by default
  - `setTargetName("_blank");`
- Does not broadcast events
- There are other ways
  - `Label` in `ContentMode.XHTML`
  - `Button` styled as link
    - Broadcasts events

COM.VAADIN.UI.MENUBAR

- Horizontal menu bar
  - With submenus and submenus...
  - `.addItem(String, Resource, MenuBar.Command)`
    → `MenuBar.MenuItem`
    - Item text
    - Item icon (optional)
    - Command executed upon clicking (optional)
    - Returned menu item can contain more items
- Just like in desktop apps
  - But positioned anywhere

vaadin}>

COM.VAADIN.UI.UPLOAD

- Allows file upload
- Broadcasts a lot of events
- Can be somewhat customised
  - Except the file browse button
- A complete example in the Book

COM.VAADIN.UI.POPUPVIEW

- A component with a surprise
  - Pop-up can hold any content
  - Minimised version can be `null` (i.e. hidden)
- Can open on click
  - Or through code
- Pop-up disappears when it loses focus
  - And broadcasts events when it shows or hides

com.vaadin.ui.Embedded

PURPOSE

- Embeds a resource
  - PDF, applet, stream...
- Three common cases excluded
  - All inherit from `AbstractEmbedded`
    - Embedded does not

com.vaadin.ui.Flash

- Embeds Flash content
  - But why would you do that?

com.vaadin.ui.Image

- Embeds an image
- Broadcasts click events

com.vaadin.ui.BrowserFrame

- Embeds web page
- Rendered as `<iframe>`

COMPONENT CONTAINERS

Components that contain other components, but are not layouts

COM.VAADIN.UI.PANEL

- Useful things
  - Caption, Icon, Border, Scrollbar
    - Scrollable programmatically
  - Broadcasts clicks
    - Unless a component inside ate it
- Can hold exactly one component
  - Though that component can contain other components which can contain further components
    - We have to go deeper
- Adds complexity to layout
  - Abuse increases page rendering time

COM.VAADIN.UI.ABSTRACTSPLITPANEL

- Holds two components
  - Which can contain components which… etc.
    - Might be `Panels` to magically allow scrollbars
- Two subclasses
  - `HorizontalSplitPanel`
    - Components arranged horizontally
    - The split is a vertical line
  - `VerticalSplitPanel`
    - The other way
- Split can be locked
  - User cannot change the position of the split

COM.VAADIN.UI.TABSHEET

- Multicomponent container
  - Each component on a separate tab
    - Caption, Icon, Description, Visibility, Availability, Ability to be closed
- Tabs are (currently) not loaded until selected
- Tab headers are arranged horizontally
  - Tab is displayed below tab headers
- One subclass, `Accordion`
  - Tab headers are arranged vertically
    - Tab is displayed between tab headers
- Events
  - Selected tab changes
    - Or when the first tab is added (it gets selected)
  - A tab is closed

BASIC FIELDS

Components that accept a simple value

OVERVIEW

- Interaction with the user
  - Contain user-assignable value
  - Broadcast events that can be listened to
    - `ClickEvent FocusEvent BlurEvent`
    - `ValueChangeEvent ReadOnlyStatusChangeEvent`
- Designed to fit the Vaadin Data Model
  - Explained later in the course
  - Can be used without a data source

COM.VAADIN.UI.BUTTON

- Clickable thingy
  - Finalises user input
  - Initialises an action
  - Can be automatically disabled on click
  - Can contain HTML caption
- Clickable in a number of ways
  - Mouse
  - Keyboard shortcut
  - Method call
- Some predefined style names in helpers

COM.VAADIN.UI.CHECKBOX

- Two-state selection component
  - Selected
  - Deselected
  - None of the above
- Two distinct roles
  - Confirms something when alone
  - Selects options when in group

com.vaadin.ui.AbstractTextField

PURPOSE

- Common class for text inputs
  - Can restrict length
  - May have an input prompt
    - Aside from caption, description and icon
- Broadcasts also `TextChangeEvents`

com.vaadin.ui.TextArea

- Supports multi-line text
- Can wrap lines

com.vaadin.ui.TextField

- Typical single-line text input field
- Can have a `Converter` to support non-string values

com.vaadin.ui.PasswordField

- Single-line text input field with hidden text
- Does not encrypt its contents

COM.VAADIN.UI.DATEFIELD

- Holds `java.time.LocalDate` as value
  - Pre-8 it is `java.util.Date`
- Two available subclasses
  - `InlineDateField`
    - Displays inline editor
  - `PopupDateField`
    - Displays input with a button
    - Button shows date picker in a popup

SELECTION FIELDS

Components that allow to select option(s) from a data source

OVERVIEW

- Advanced interaction with the user
  - Broadcast events that can be listened to
    - `ClickEvent FocusEvent BlurEvent`
    - `ValueChangeEvent SelectionChangeEvent`
- Contain user-assignable value
  - Selectable from a range of options
    - `setItems()`
    - `setContainerDataSource()`
  - Fully configurable display
    - `setItemCaptionProvider()`

COM.VAADIN.UI.COMBOBOX

- Selects a single value
- Allows creating new values
- Dynamic item suggestion
    - As the user types in
- Default choice for a drop-down component

COM.VAADIN.UI.NATIVESELECT

- Selects a single value
- As simple as it can be
  - Browser native
  - No new values
  - No dynamic item suggestion

COM.VAADIN.UI.LISTSELECT

- Can select multiple values
- As simple as it can be
  - Browser native
  - No new values
  - No dynamic item suggestion

COM.VAADIN.UI.OPTIONGROUP

- Can select multiple values
  - Group of radio buttons in single-selection mode
  - Group of check boxes in multi-selection mode
- Supports disabling individual items

COM.VAADIN.UI.
TWINCOLSELECT

- Multiple selection based on two columns
  - *Options* and *selections*
- Some customisation
  - Captions for each column
  - Number of rows visible

OTHER COMPONENTS

Moar components!

Data presentation components

com.vaadin.ui.Table

- Tabular representation
- Lots of options
  - Headers, footers
  - Multiselect or no selection at all
  - Hidden columns
  - Moving columns
  - Sorting
  - In-place editing
  - Cell generators
  - Drag and drop
- Lots of code
  - I mean, really
- Subclassed by `TreeTable`

com.vaadin.ui.Tree

- Tree representation
  - Hierarchy
- Options
  - Multiselect
  - Drag and drop
  - Node collapsing
    - With custom icons

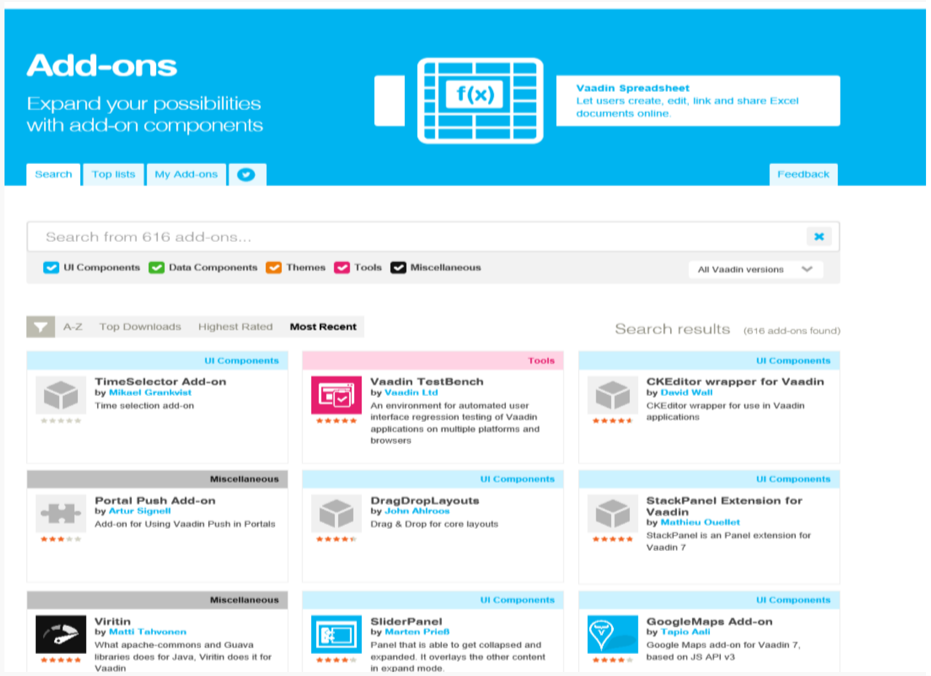
COM.VAADIN.UI.GRID

- `Table` on steroids
  - Lazy loading
  - Header, footer, sorting, reordering, freezing…
  - Custom row generation
    - But no custom column generator
- Not a `Field`
  - For reasons that I have yet to understand
  - Custom selection and value handling
- Not a replacement for `Table`
  - Both components have their use cases

...AND EVEN MORE

- `com.vaadin.ui.ProgressBar`
  - Displays progress as a value from 0.0 to 1.0
  - Browser can poll or server can push
- `com.vaadin.ui.Slider`
  - For selecting numerical value within a range
  - Can be horizontal or vertical

HTTPS://VAADIN.COM/DIRECTORY



Add-ons

Expand your possibilities with add-on components

Vaadin Spreadsheet
Let users create, edit, link and share Excel documents online.

Search | Top lists | My Add-ons | Feedback

Search from 616 add-ons...

UI Components | Data Components | Themes | Tools | Miscellaneous | All Vaadin versions

A-Z | Top Downloads | Highest Rated | **Most Recent**

Search results (616 add-ons found)

UI Components
TimeSelector Add-on
by Mikael Grankvist
Time selection add-on

Tools
Vaadin TestBench
by Vaadin Ltd
An environment for automated user interface regression testing of Vaadin applications on multiple platforms and browsers

UI Components
CKEditor wrapper for Vaadin
by David Wall
CKEditor wrapper for use in Vaadin applications

Miscellaneous
Portal Push Add-on
by Artur Signell
Add-on for Using Vaadin Push in Portals

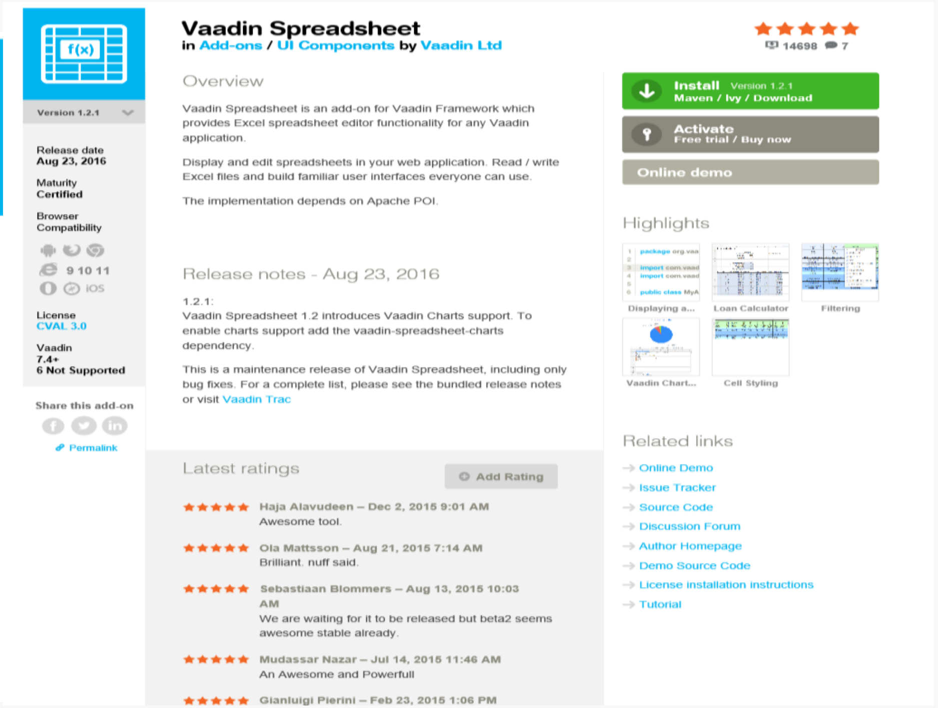
UI Components
DragDropLayouts
by John Ahlroos
Drag & Drop for core layouts

UI Components
StackPanel Extension for Vaadin
by Mathieu Ouellet
StackPanel is an Panel extension for Vaadin 7

Miscellaneous
Viritin
by Matti Tahvonen
What apache-commons and Guava libraries does for Java, Viritin does it for Vaadin

UI Components
SliderPanel
by Marten Prieß
Panel that is able to get collapsed and expanded. It overlays the other content in expand mode.

UI Components
GoogleMaps Add-on
by Tapio Aali
Google Maps add-on for Vaadin 7, based on JS API v3



Vaadin Spreadsheet
in Add-ons / UI Components by Vaadin Ltd

14698 7

Version 1.2.1

Release date
Aug 23, 2016

Maturity
Certified

Browser Compatibility
9 10 11
iOS

License
CVAL 3.0

Vaadin
7.4+
6 Not Supported

Share this add-on

Permalink

Overview

Vaadin Spreadsheet is an add-on for Vaadin Framework which provides Excel spreadsheet editor functionality for any Vaadin application.

Display and edit spreadsheets in your web application. Read / write Excel files and build familiar user interfaces everyone can use.

The implementation depends on Apache POI.

Release notes - Aug 23, 2016

1.2.1:
Vaadin Spreadsheet 1.2 introduces Vaadin Charts support. To enable charts support add the vaadin-spreadsheet-charts dependency.

This is a maintenance release of Vaadin Spreadsheet, including only bug fixes. For a complete list, please see the bundled release notes or visit Vaadin Trac

Latest ratings

Add Rating

Haja Alavudeen – Dec 2, 2015 9:01 AM
Awesome tool.

Ola Mattsson – Aug 21, 2015 7:14 AM
Brilliant. nuff said.

Sebastiaan Blommers – Aug 13, 2015 10:03 AM
We are waiting for it to be released but beta2 seems awesome stable already.

Mudassar Nazar – Jul 14, 2015 11:46 AM
An Awesome and Powerfull

Gianluigi Pierini – Feb 23, 2015 1:06 PM

Install Version 1.2.1
Maven / Ivy / Download

Activate
Free trial / Buy now

Online demo

Highlights

Displaying a... | Loan Calculator | Filtering | Vaadin Chart... | Cell Styling

Related links

- Online Demo
- Issue Tracker
- Source Code
- Discussion Forum
- Author Homepage
- Demo Source Code
- License installation instructions
- Tutorial

DEMO!

Shoutbox step ~~1~~ 2
http://github.com/vaadin-miki/shoutbox

end branch: `step-02`

CHOOSING COMPONENTS

- Text field for user text
- Button for submitting
- Labels for messages
- Basic vertical layout
  - More on layouts soon
- One event – button click
  - More on events soon

EXECUTION

- `mvn jetty:run`
  - Right click project → Run as Maven build…
  - http://localhost:8080
- Widgetset compilation?
  - Client-side JS for components
    - Based on GWT
  - Moved to cloud

    -
      ```
      <configuration>
        <widgetsetMode>cdn</widgetsetMode>
       </configuration>
      ```
    - Startup time reduced from 16 seconds to 1.6 second
      - On a quite old machine, mind you

SUMMARY

What did we do today?

LESSONS OF TODAY (HOPEFULLY)

- Component hierarchy
  - What are the basic interfaces?
  - What are the differences?
- Common properties
  - What are they?
  - Where are they defined?
- Components
  - How are they grouped?
  - What events do they broadcast and when?
- Coding
  - How to get rid of widgetset compilation ?

COMING UP NEXT

- Events and data binding
- Styling, theming, layouts, navigation

THE END

SUGGESTIONS?
QUESTIONS?

miki@vaadin.com
t: @mikiolsz