

Mobile Application Development

Higher Diploma in Science in Computer Science

Produced
by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>

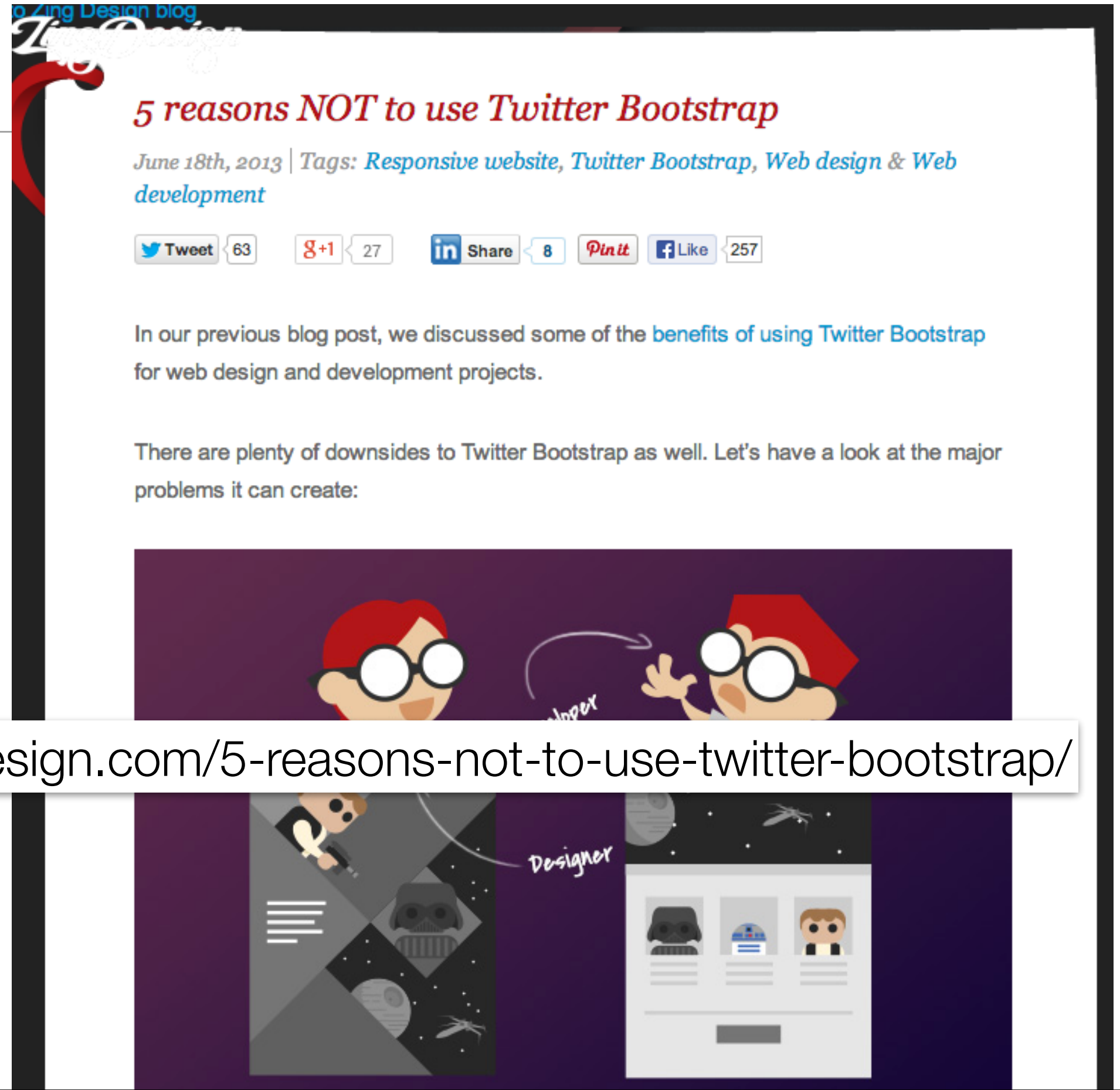


Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE



Semantic UI

The Trouble with Bootstrap...



<http://www.zingdesign.com/5-reasons-not-to-use-twitter-bootstrap/>

1. It doesn't follow best practices

- ... you end up with a whole lot of DOM elements crammed full of classes.
- ... breaks one of the fundamental rules of good web design, the HTML is no longer **semantic** and the presentation is no longer separate from the content.
- ... makes scalability, reusability and maintenance that much more of a challenge
- ...also exacerbates progressive enhancement as presentation and interaction are no longer independent of content.

2. It's going to collide with my existing set-up

- ... lot of problems ... conflicts in generated HTML, CSS and JavaScript to begin with
- ... have to go through that big monster of a project and work out which scripts and styles need to be removed or replaced.
- ... could potentially create extra work as you go through the project inevitably finding and fixing weird bugs
- ... defeats the purpose of using it in the first place.

3. Twitter Bootstrap is heavy

- .. includes CSS weighing in at 126KB and 29KB of JavaScript.
- so consider your target market ... Twitter Bootstrap will help you to build an attractive, responsive website with all the bells and whistles,
- .. but some mobile users could be turned away by the slow loading time and battery-draining scripting.

4. No SASS support

- ... it's built with Less and provides no native support for Compass and SASS.
- ... SASS is just better, and with a framework like Compass on top, it seems like a complete no-brainer to use it.
- Some helpful folks have built a Bootstrap for Compass gem, but straight out of the box, you'll have to make do with Less.

5. “Hey! My new website looks just like everyone else’s!”

- While it is possible to customise your app or website design further, you may find time constraints force you to stick to a lot of the vanilla Bootstrap style.
- This can lead to the inadvertent creation of a lot of similar, generic and unmemorable websites.
- While Twitter Bootstrap is fast and easy to implement, creativity is often compromised as a result.
- Innovative designs which defy conventions can be difficult to implement in Bootstrap’s structured environment.

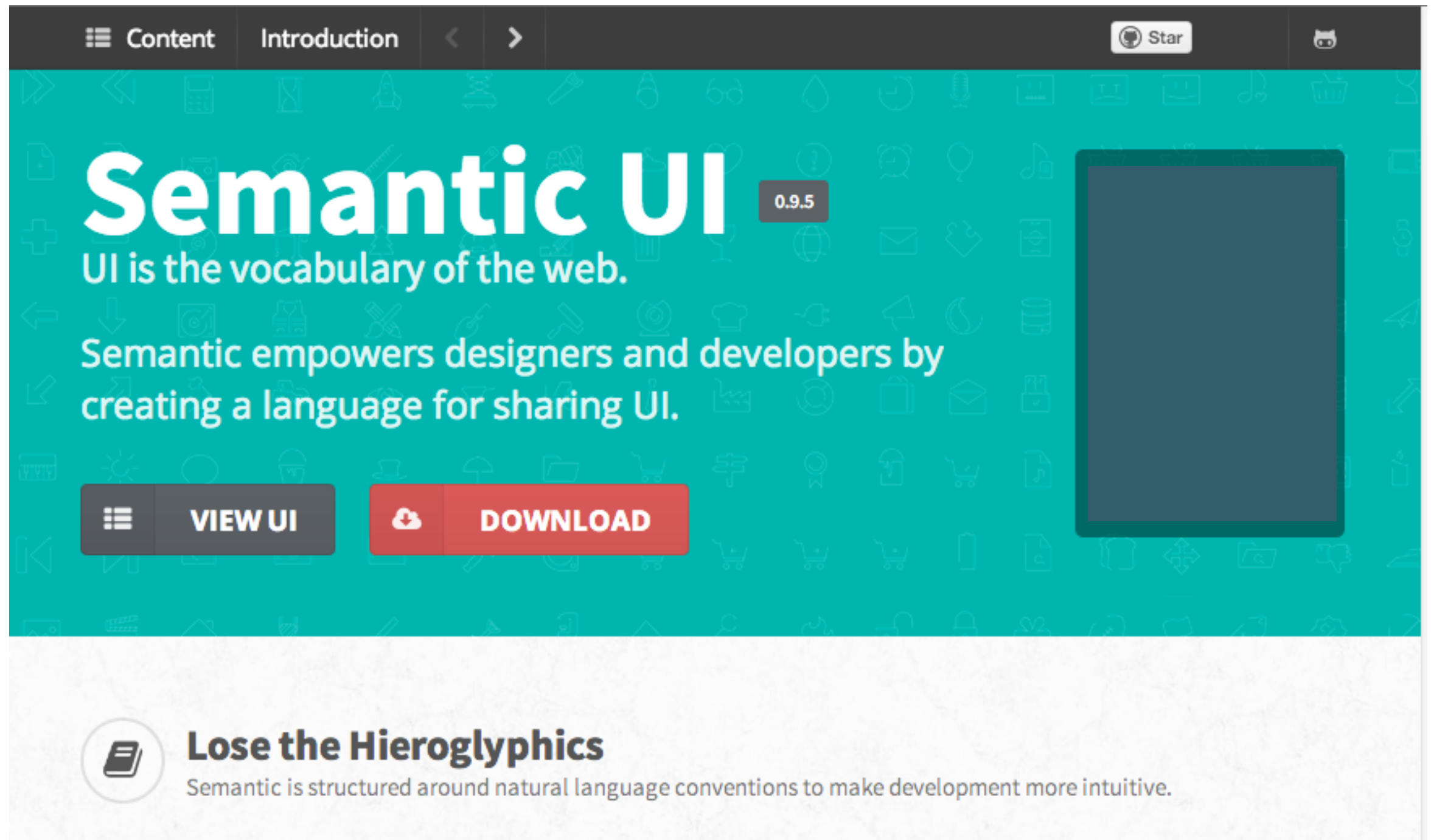
Bootstrap not Semantic

- The html pages become cluttered with presentation detail - difficult to maintain and challenging to redesign

*“... breaks one of the fundamental rules of good web design, the HTML is no longer **semantic** and the presentation is no longer separate from the content.”*

```
<hr class="featurette-divider">
▼<div class="featurette">
  
  ▶<h2 class="featurette-heading">...</h2>
  ▶<p class="lead">...</p>
</div>
<hr class="featurette-divider">
▼<div class="featurette">
  
  ▶<h2 class="featurette-heading">...</h2>
  ▶<p class="lead">...</p>
</div>
```

Semantic UI



<http://semantic-ui.com/>

SEMANTIC

```
<main class="ui three column grid">
  <aside class="column">1</aside>
  <section class="column">2</section>
  <section class="column">3</section>
</main>
```

BOOTSTRAP

```
<div class="row">
  <div class="col-lg-4">1</div>
  <div class="col-lg-4">2</div>
  <div class="col-lg-4">3</div>
</div>
```

SEMANTIC

```
<nav class="ui menu">
  <h3 class="header item">Title</h3>
  <a class="active item">Home</a>
  <a class="item">Link</a>
  <a class="item">Link</a>
  <span class="right floated text item">
    Signed in as <a href="#">user</a>
  </span>
</nav>
```

BOOTSTRAP

```
<div class="navbar">
  <a class="navbar-brand" href="#">Title
</a>
  <ul class="nav navbar-nav">
    <li class="active"><a href="#">Home
</a></li>
    <li><a href="#">Link</a></li>
    <li><a href="#">Link</a></li>
    <p class="navbar-text pull-right">
      Signed in as <a href="#" class="navbar
-link">User</a></p>
  </ul>
</div>
```

SEMANTIC

```
<button class="large ui button">
  <i class="heart icon"></i>
  Like it
</button>
```

BOOTSTRAP

```
<button type="button" class="btn btn
-primary btn-lg">
  <span class="glyphicon glyphicon-heart"
></span>
  Like
</button>
```

Defining Definitions

Definitions in semantic are groups of css, fonts, images, and javascript which make up a single element. Unlike other javascript libraries, semantic UI elements are stand-alone and only require their own assets to function correctly.

Definition Types

Semantic has five different ui definitions. These are useful patterns for describing re-usable parts of a website.



UI Element	A basic building block of a website, exists alone or in homogenous groups
UI Collection	A heterogeneous group of several elements which can usually be found together.
UI View	A way to present common website content like comments, activity feeds
UI Module	An element where its behavior is an essential part of its definition
UI Behavior	A set of free-standing actions not specifically tied to an element

Definition Sections



- All UI components of a single type are defined similarly

All UI	Types Defines mutually exclusive types which each may have their own html
	States Defines element states like disabled, hovered, pressed down
	Variations Defines changes to an element which are not mutually exclusive and can be used together

Elements & Collections

Elements	<div data-bbox="617 598 765 649">Group</div> <div data-bbox="617 664 2362 715">An element can optionally define how attributes can be shared across a group</div> <div data-bbox="617 766 913 815"> UI Button</div>
Collections	<div data-bbox="617 917 806 966">Content</div> <div data-bbox="617 983 1983 1032">A collection can define elements which might be found inside</div> <div data-bbox="617 1085 762 1134">States</div> <div data-bbox="617 1150 1942 1199">A collection may define states for content elements or itself</div> <div data-bbox="617 1252 858 1302">Variations</div> <div data-bbox="617 1318 2030 1367">A collection may define variations for content elements or itself</div> <div data-bbox="617 1420 872 1469"> UI Form</div>

Views, Modules & Behaviours

Views	<div data-bbox="644 562 836 613">Content</div> <div data-bbox="644 623 2030 684">A view may define elements which can exist inside of the view</div> <div data-bbox="644 725 795 776">States</div> <div data-bbox="644 786 1865 848">A view may define states for content elements or itself</div> <div data-bbox="644 889 891 940">Variations</div> <div data-bbox="644 950 1989 1011">A view may define variations for a content elements or itself</div> <div data-bbox="644 1052 891 1113"> UI Item</div>
Modules & Behaviors	<div data-bbox="644 1216 864 1267">Behavior</div> <div data-bbox="644 1277 2154 1338">A module will define a set of behaviors which can be used as an API</div> <div data-bbox="644 1379 836 1430">Settings</div> <div data-bbox="644 1441 2483 1502">A settings object which can alter the default behavior when instantiating a module</div> <div data-bbox="644 1543 878 1594">Examples</div> <div data-bbox="644 1604 2181 1665">A list of examples to showcase the variations in behavior of a module</div> <div data-bbox="644 1706 1029 1768"> UI Dropdown</div>

Types Catalogue

12

UI Elements

Button

Divider

Header

Icon

Image

Input

Label

Loader

Progress

Reveal

Segment

Step

6

UI Collections

Breadcrumb

Form

Grid

Menu

Message

Table

4

UI Views

Comment

Feed

Item

List

11

UI Modules

Accordion

Checkbox

Dimmer

Dropdown

Modal

Popup

Rating

Shape

Sidebar

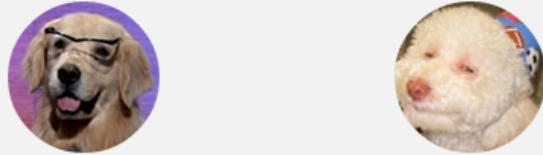
Transition

Validate Form

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante.

Segment

A segment is used to create a grouping of related content.



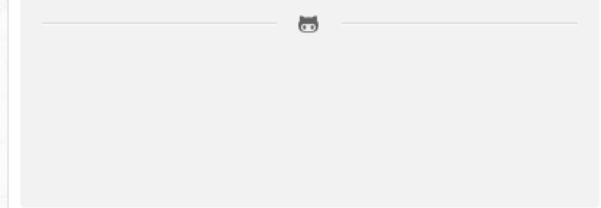
Image

An image is a graphic representation of something



Button

A button indicates a possible user action.



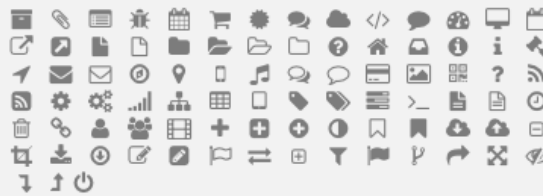
Divider

A divider visually segments content into separate groups



Step

Steps show the current activity in a series of steps.



Icon

An icon is a glyph used to represent another concept more simply

Section 2

The second section of the website

Header

Headers provide a short summary of content



Progress Bar

A progress bar displays progress on a task

DOG

HTML

Label

Labels give descriptions to sections of content.

Elements

Collections


Food / Fruit / Apples

Food > Fruit > Apples

Breadcrumb

A breadcrumb is a menu to show the location of the current section in relation to other sections.

Name

 Name

E-mail

E-mail

Form

A form is used to solicit a user response

1

2

3

Grid

A grid helps harmonize negative space in a layout

Friends

Messages

Profile

Friends

Messages

Profile

Menu

A menu organizes related links

**We're sorry we can't
process your idea just
yet**

Please enter your name

Message

Messages alert a user to something important.

Name

Status

John

Approved

John

Unconfirmed

Sally

Denied

Table

A table collects related data into rows of content

Modules

Overview

Initializing

Behaviors

Settings

All official javascript modules in Semantic are designed using a singular design pattern. This pattern allows several useful features common to all javascript components

✓ **Run-time Performance Analysis**

Semantic modules all provide the ability to log performance traces to the console, allowing you to see which aspects of the module are more or less performant and to track total init time `onDomReady`

✓ **Human Readable Traces**

Unlike other component libraries which hides explanations of behavior in inline comments which can only be read by combing the source, semantic modules provide run-time debug output to the javascript console telling you what each component is doing as it is doing it.

✓ **Settings can be overwritten after initialization**

Semantic provides methods to set default settings, set settings at initialization, and set settings after initialization, allowing complete flexibility over component behaviors.

✓ **All modules include an initialize and destroy method**

All events and metadata are namespaced and can be removed after initialization, modules automatically handle destroy/init events to allow users to lazy-initialize a plugin multiple times with no issues.

✓ **Instance available in metadata**

Modules store their instance in metadata meaning that, in a pinch, you can directly modify the instance of a UI element by modifying its properties.



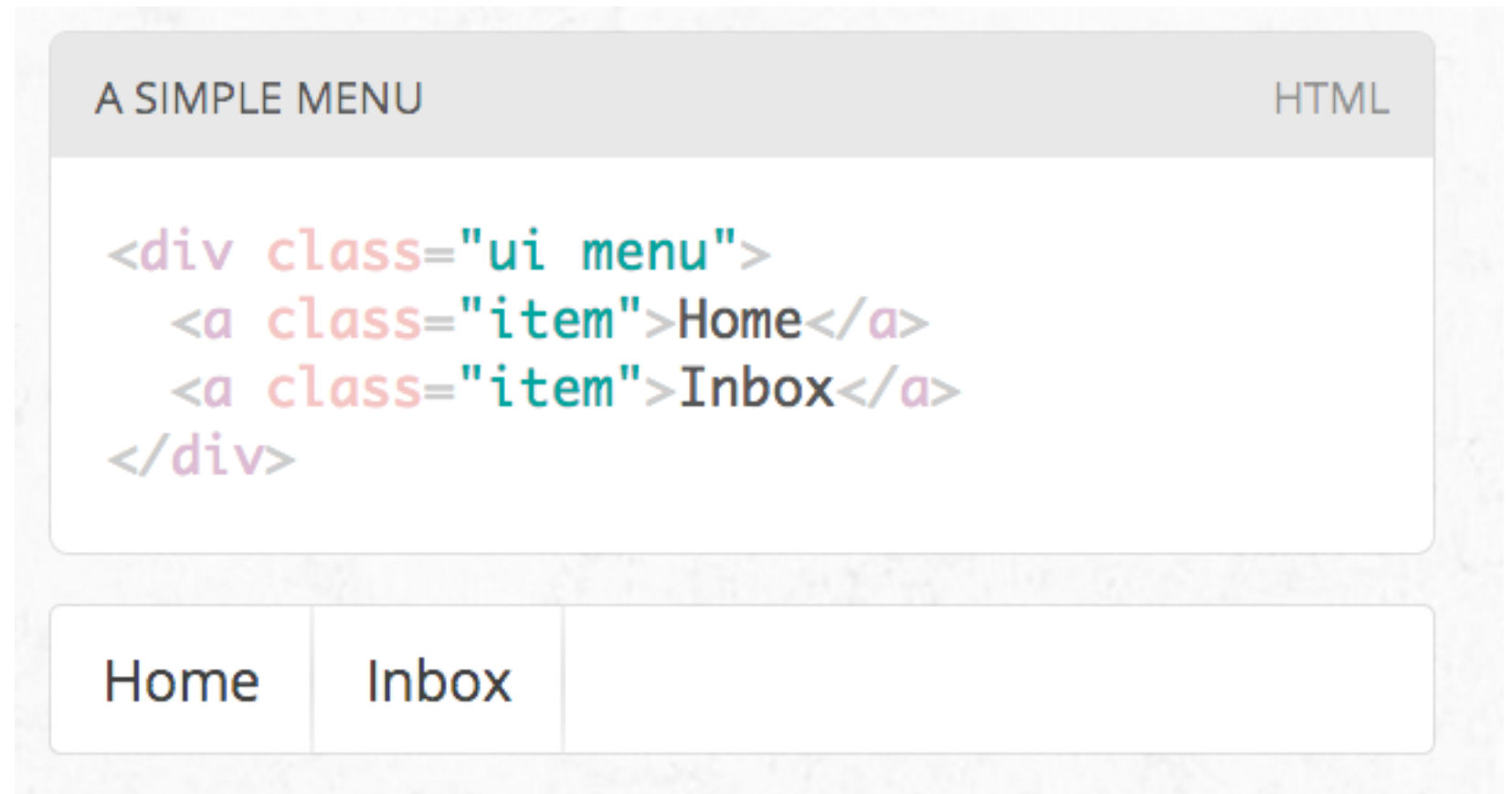
REAL WORLD COMMENTED EXAMPLE



COMMENTED DESIGN PATTERN

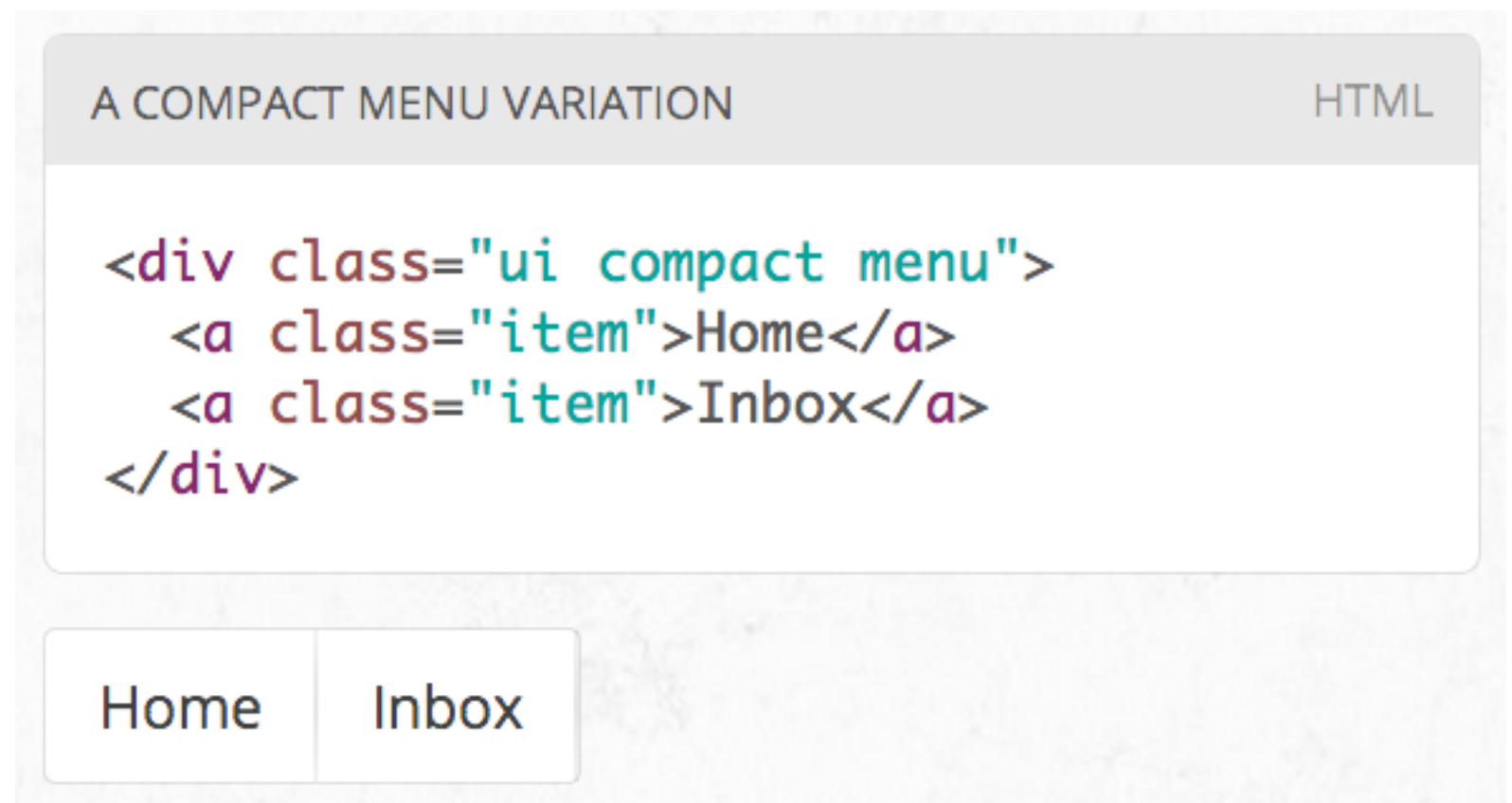
Interfacing Carefully

- UI definitions in Semantic are given the class name ui.
- This is to help tell the difference between ui elements and parts of the definition of an element.
- This means any element with the class name UI has a corresponding UI definition.



Changing an Element

- Class names in Semantic always use single english words.
- If a class name is an adjective it is either a type of element or variation of an element.
- CSS definitions always define adjectives in the context of a noun. In this way class names cannot pollute the namespace.



Combining an Element

- All UI definitions in semantic are stand-alone, and do not require other components to function.
- However, components can choose to have optional couplings with other components.
- For example you might want to include a badge inside a menu. A label inside of a menu will automatically function as a badge

USING A UI LABEL INSIDE A UI MENU

HTML

```
<div class="ui compact menu">  
  <a class="item">Home</a>  
  <a class="item">  
    Inbox  
    <div class="ui label">22</div>  
  </a>  
</div>
```

Home

Inbox

22


Types / Variations


- A ui definition in Semantic usually contains a list of mutually exclusive variations on an element design.
- A type is designated by an additional class name on a UI element


TYPES OF UI BUTTON

HTML

```
<div class="ui labeled icon button">
  Download <i class="download icon"></i>
</div>
<div class="ui icon button">
  <i class="download icon"></i>
</div>
<div class="ui button">
  Download
</div>
<div class="ui facebook button">
  <i class="facebook icon"></i>
  Facebook
</div>
```

 **DOWNLOAD**

 **DOWNLOAD**

 **FACEBOOK**

Types / Content

- Types may require different html structures to work correctly.
- For example, an icon menu might expect different content like icons glyphs instead of text to be formatted correctly

ICON MENU TYPE

HTML

```
<div class="ui icon menu">
  <a class="item">
    <i class="mail icon"></i>
  </a>
  <a class="item">
    <i class="lab icon"></i>
  </a>
  <a class="item">
    <i class="star icon"></i>
  </a>
</div>
```





Types / HTML Differences

- Types may also each require slightly different html.
- For example, a tiered menu needs html specified for a sub menu to display itself correctly

TIERED MENU TYPEHTML

```
<div class="ui tiered menu">
  <div class="menu">
    <div class="active item">
      <i class="home icon"></i>
      Home
    </div>
    <a class="item">
      <i class="mail icon"></i>
      Mail
      <span class="ui label">22</span>
    </a>
  </div>
  <div class="sub menu">
    <div class="active item">Activity</div>
    <a class="item">Profile</a>
  </div>
</div>
```

 Home

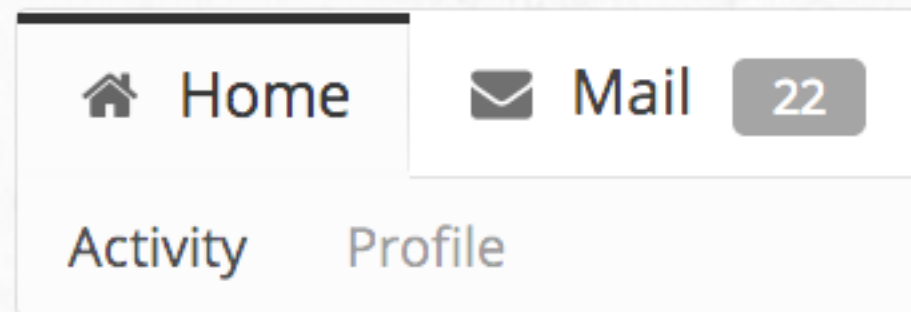
 Mail 22

Activity Profile

Variations

- A variation alters the design of an element but is not mutually exclusive.
- Variations can be stacked together, or be used along with altering an element's type.
- For example, having wide menus that take up the full width of its parent may sometimes be overwhelming. You can use the compact variation of a menu to alter its format to only take up the necessary space.

```
<div class="ui compact tiered menu">  
  ...  
</div>
```



Intersecting Variations

- The definition for the variation red contains css specifically for describing the intersection of both red and inverted.

```
<div class="ui red tiered menu">  
  ...  
</div>
```



Home



Mail

22

Activity

Profile

Whats Different?

Build Responsive Layouts Easier

- Designed Completely with EM
- Every component is defined using em and rem so that components can be resized simply on the fly.
- Want a menu to get smaller on mobile? Simply have it's font-size change using a media query.

Self Explanatory

- Descriptive not Prescriptive
- Writing front end code shouldn't require learning the naming or programming conventions of a particular developer.
- Instead of using short-hand, or codifying naming conventions, Semantic uses simple, common language for parts of interface elements, and familiar patterns found in natural languages for describing elements.

Tag ambivalent

- Use whatever html tags you please.
- Interface definitions in Semantic are tag ambivalent.
- That means you can use div, article, section, span without affecting the display of the element.
- Special tags like a, table, td still carry special meaning in certain circumstances however.

Powerful tools for expressing groups and collections.

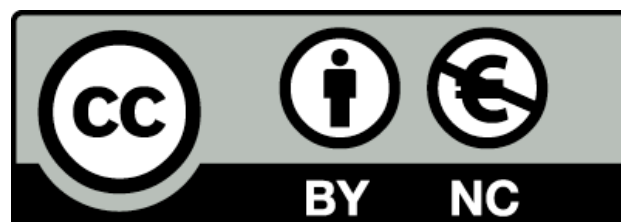
- Don't repeat yourself
- In English it's much easier to say "There are three tall men" than "There is a tall man, a tall man and a tall man".
- In Semantic element definitions can be expressed in groups have shared attributes like size, color, type avoiding repetitive declarations.

Portable and self-contained.

- Using Semantic doesn't mean adopting an entire framework, or rewriting your code base
- Semantic components are written in a singular style, but are not part of mandated overarching library. Only like a couple components? No problem, use only what you need.
- UI components in Semantic also define optional and required couplings with other components where their usage intersect. That means for example, a popup can check for the existence of CSS animation component before using the fallback javascript animations.

Shared language, different implementations

- Restyle your site without Restructuring it
- Describing your site's content using a common language like Semantic allows other developers to create UI definitions to match one shared vocabulary.
- This means you can redesign your website without retooling your html.
- Simply alter the look and feel of your UI using a different UI style definition.



Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

