

Advanced Programming 2

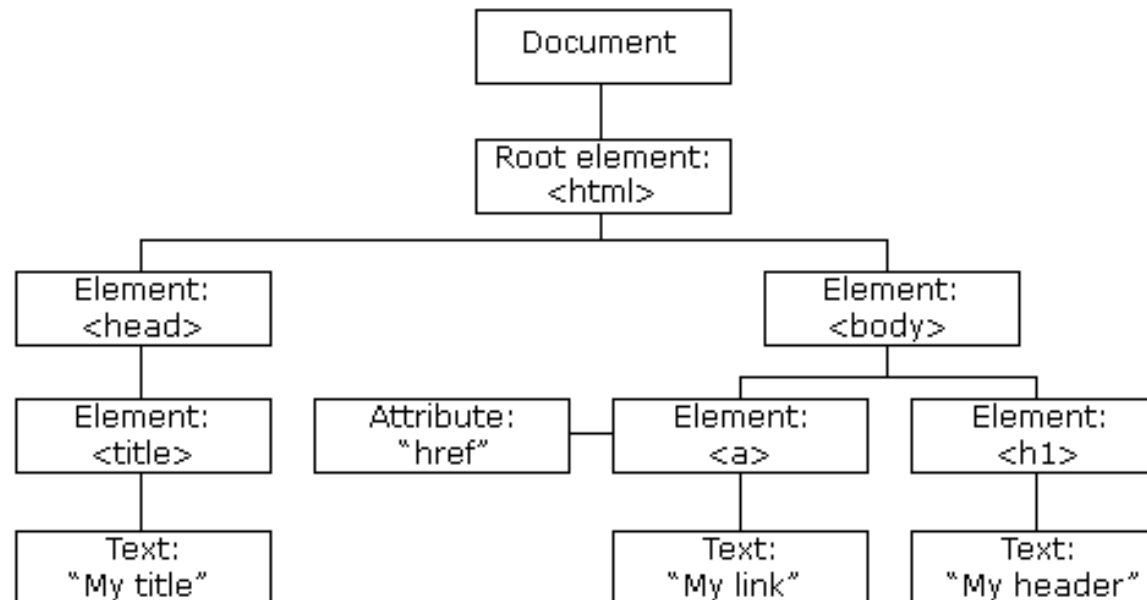
Recitation 9 – Web Applications Part III

Roi Yehoshua
2017

The HTML DOM

The HTML DOM

- ▶ The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document.
- ▶ When a web page is loaded, the browser creates in memory the DOM representation of the page, which is constructed as a tree of objects:

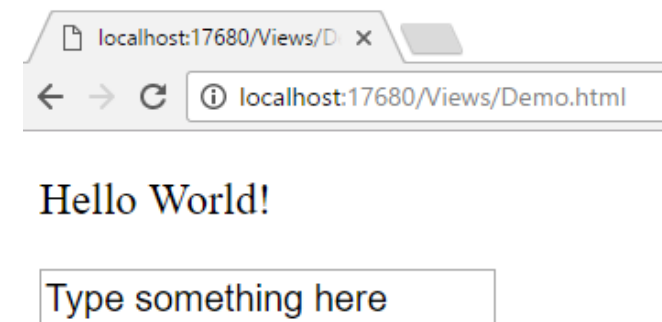


The HTML DOM

- ▶ In the DOM, all HTML elements are defined as **objects**
- ▶ The programming interface is the properties and methods of each object
- ▶ A **property** is a value that you can get or set (like changing the content of an HTML element)
- ▶ A **method** is an action you can do (like add or deleting an HTML element)
- ▶ The most common way to access an HTML element is to use the id of the element using the **getElementById()** method

```
<html>
<body>
  <p id="demo"></p>
  <input type="text" id="txt1"/>
  <script>
    var p = document.getElementById("demo")
    p.innerHTML = "Hello World!";

    var txt1 = document.getElementById("txt1");
    txt1.value = "Type something here";
  </script>
</body>
</html>
```



Events

- ▶ Events are properties of DOM objects
- ▶ Events are generated by the browser when "things happen" to HTML elements, e.g.:
 - ▶ An element is clicked on
 - ▶ The page has loaded
 - ▶ Input fields are changed
- ▶ You can assign an event to a function
 - ▶ Typically an anonymous function

```
<html>
<body>
  <h1 id="id1">My Heading</h1>
  <button id="btn1">Click Me!</button>

  <script>
    var btn1 = document.getElementById("btn1");
    btn1.onclick = function () {
document.getElementById("id1").style.color = 'red'
    };
  </script>
</body>
</html>
```

← → ↻ ⓘ localhost:17680/Views/Demo.html

My Heading

Click Me!

jQuery

Getting Started

- ▶ Cross-browser, JavaScript library designed to simplify client-side scripting
- ▶ Used by over 50% of the 10,000 most visited web sites
- ▶ jQuery is free, open source software
- ▶ Downloadable from <http://www.jquery.com>
- ▶ Main features:
 - ▶ DOM element selections and modifications
 - ▶ Events Handling
 - ▶ Effects and animations
 - ▶ Ajax
 - ▶ Extensibility through plug-ins
 - ▶ Cross-browser support

jQuery Syntax

- ▶ Basic syntax is: **`$("selector").action()`**
 - ▶ The `$` sign stands for the jQuery function
 - ▶ A *selector* is used to query or find HTML elements
 - ▶ All CSS selectors are supported (including CSS3 selectors)
 - ▶ `$("selector")` creates a new jQuery object that contains the selected elements
 - ▶ The jQuery *action()* is performed on the selected element(s)
- ▶ Examples:
 - ▶ `$("p").hide()` - hides all `<p>` elements
 - ▶ `$(".test").hide()` - hides all elements with `class="test"`
 - ▶ `$("#test").hide()` - hides the element with `id="test"`
 - ▶ `$(this).hide()` - hides the current element

jQuery DOM Manipulation

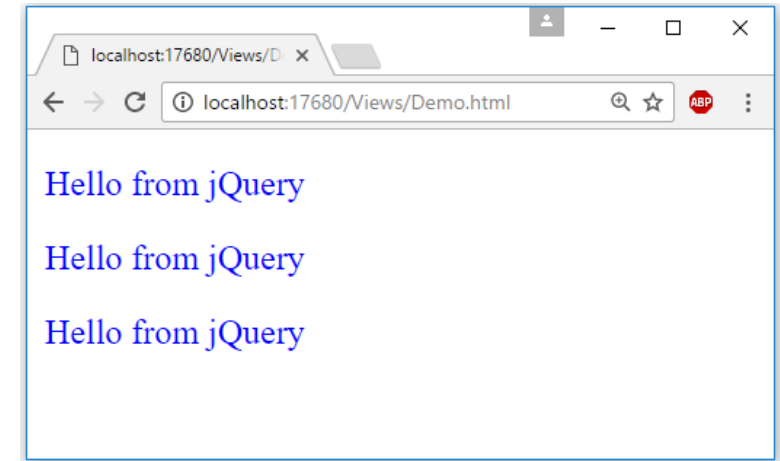
- ▶ jQuery provides a set of methods to access and manipulate DOM elements and attributes:
 - ▶ **text()** - sets or returns the text content of selected elements
 - ▶ **html()** - sets or returns the content of selected elements (including HTML markup)
 - ▶ **val()** - sets or returns the value of form fields
 - ▶ **attr()** - sets or returns HTML attribute values
 - ▶ **css()** - sets or returns the style attribute

jQuery Example

```
<!DOCTYPE html>
<html>
<head>
  <title>jQuery Example</title>
</head>
<body>
  <p>First paragraph</p>
  <p>Second paragraph</p>
  <p>Third paragraph</p>

  <script src="jquery-3.2.1.js"></script>
  <script>
    $("p").html("Hello from jQuery").css("color", "blue");
  </script>
</body>
</html>
```

Note the method chaining.
This is a general feature of jQuery.



DOM Traversal

- ▶ jQuery provides methods for selecting the **children**, the **parents**, and the **siblings** of each element in the wrapped set
- ▶ Most of these methods accept an option jQuery selector argument for **filtering**
- ▶ The value **returned** by these methods is a **jQuery object**

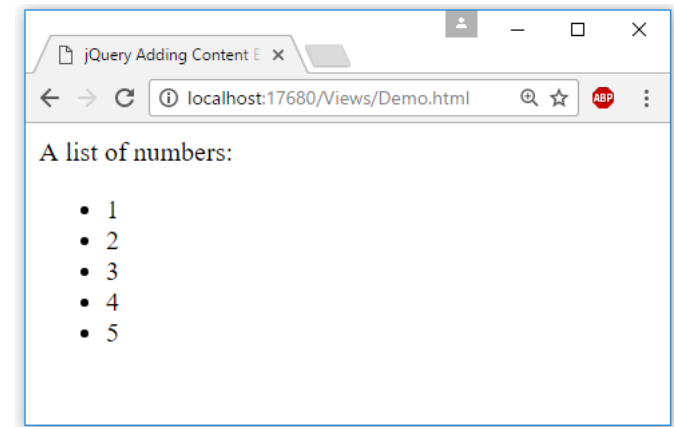
```
$("#div").children("p") // Select p elements that are direct descendants of div elements
$("#div").find("p")     // Select p elements that are descendants of div elements
$("#div").prev()        // prev(elem)Select set of elements immediately precede div
                        elements
$("#div").next()         // next(elem)Select set of elements immediately follow div elements
$("#div").siblings("p") // Select p elements that are siblings of div elements
$("#div").parent()       // Select elements that are parents of div elements
```

Add New Elements

- ▶ There are four jQuery methods that are used to add new content:
 - ▶ **append()** - Inserts content at the end of the selected elements
 - ▶ **prepend()** - Inserts content at the beginning of the selected elements
 - ▶ **after()** - Inserts content after the selected elements
 - ▶ **before()** - Inserts content before the selected elements

```
<body>
  A list of numbers:
  <ul id="lstNumbers"></ul>

  <script src="jquery-3.2.1.js"></script>
  <script>
    for (var i = 1; i <= 5; i++) {
      $("#lstNumbers").append("<li>" + i + "</li>");
    }
  </script>
</body>
```



- ▶ There is a more efficient way to build this list in jQuery (can you think of one?)

jQuery Events

- ▶ Most DOM events have an equivalent jQuery method
 - ▶ e.g., `click()`, `mouseover()`, `keydown()`
- ▶ The event handling function can access the element that the handler is bound to via the keyword **this**
- ▶ The **on()** method attaches one or more event handlers for the selected elements
 - ▶ Used for events with no equivalent jQuery methods
- ▶ The **off()** method is used to remove an event handler

```
// Assign a click event to all paragraphs
$("p").click(function() {
    $(this).hide();
});
```

```
$("p").on("click", function() {
    $(this).hide();
});
```

```
$("p").off(); // removes all event handlers from all paragraphs
$("p").off("click"); // removes all click handlers from all paragraphs
$("p").off("click", func); // func will no longer be called
```

The Event Object

- ▶ Every event handler receives an event object
- ▶ This object contains many useful properties and methods, e.g.:

Property	Description
target	The DOM element that triggered the event
pageX, pageY	Position of the mouse
which	Button or key that was pressed
preventDefault()	Prevents the default action of the event

```
<input type="text" id="txt1"/>
<script>
    $("#txt1").keydown(function (e) {
        // Handle the arrow keys
        switch (e.which) {
            case 37:
                alert("left");
                break;
            case 38:
                alert("up");
                break;
            case 39:
                alert("right");
                break;
            case 40:
                alert("down");
                break;
        }
    });
</script>
```

jQuery Plugins

- ▶ A plug-in is piece of code written in a standard JavaScript file
- ▶ Plugins provide useful jQuery methods/components which can be used along with jQuery library methods
- ▶ You could consider each method that comes with the jQuery core a plugin
- ▶ There are plenty of jQuery plug-in available which you can download from <https://jquery.com/plugins>
- ▶ Writing your own custom plugins allows you to:
 - ▶ Encapsulate your code for reuse purposes
 - ▶ Public distribution
 - ▶ Maintain chainability
 - ▶ Prevent namespace clashing

Creating a Custom Plugin

- ▶ A plugin is defined by adding a method to the jQuery prototype (jQuery.fn)
- ▶ Encapsulate the plugin in a closure
 - ▶ To hide inner functions
- ▶ The plugin is passed the jQuery object that contains the DOM selection
 - ▶ It is accessible using the **this** keyword (not \$(this))
- ▶ Return this to maintain chainability
- ▶ Save into a JS file jquery.pluginname.js
- ▶ Finally to use your plugin you would write:

```
(function ($) {  
    $.fn.pluginname = function () {  
        // Implement your plugin here  
        return this;  
    };  
})(jQuery);
```

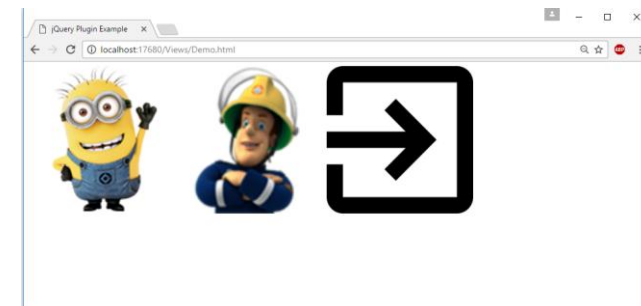
```
$("selector").pluginname();
```


Example – Make Equal Size Plugin

jQuery.makeEqualSize.js

```
(function ($) {  
    $.fn.makeEqualSize = function () {  
        var maxHeight = 0;  
        var maxWidth = 0;  
  
        this.each(function () {  
            var currHeight = $(this).height();  
            var currWidth = $(this).width();  
  
            if (currHeight > maxHeight)  
                maxHeight = currHeight;  
            if (currWidth > maxWidth)  
                maxWidth = currWidth;  
        });  
  
        this.each(function () {  
            $(this).height(maxHeight);  
            $(this).width(maxWidth);  
        });  
        return this;  
    };  
})(jQuery);
```

```
<html>  
<head>  
    <title>jQuery Plugin Example</title>  
</head>  
<body>  
      
      
      
    <script src="jquery-3.2.1.js"></script>  
    <script src="scripts/jquery.makeEqualSize.js"></script>  
    <script>  
        $("img").makeEqualSize();  
    </script>  
</body>  
</html>
```



HTML5 API

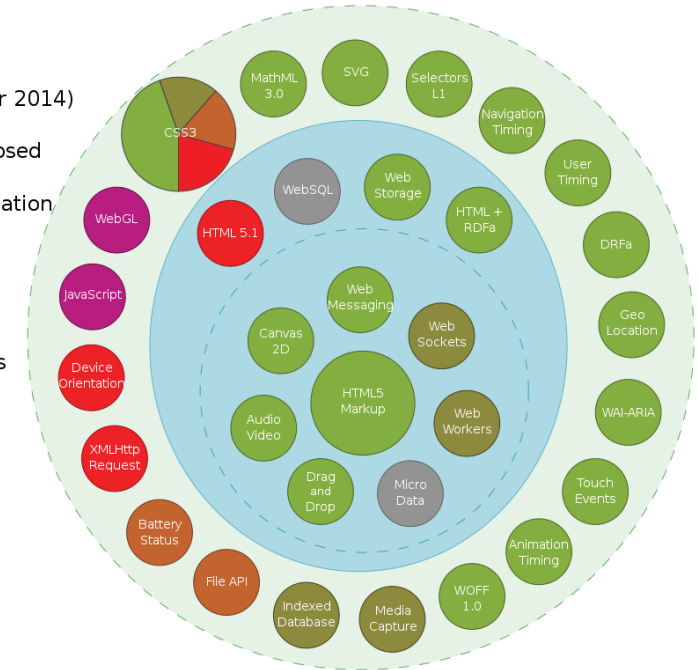
HTML5

- ▶ Goals:
 - ▶ Give better support to interactive web applications
 - ▶ Reduce the need for installing external plugins
 - ▶ like Flash or Silverlight
 - ▶ Device independence
 - ▶ More done with less code
 - ▶ The semantic web
- ▶ New APIs:
 - ▶ Canvas - dynamic rendering of 2D shapes
 - ▶ Geolocation – retrieve user's location
 - ▶ LocalStorage – storing data locally
 - ▶ IndexedDB - an indexed hierarchical key-value store
 - ▶ Drag & Drop
 - ▶ Web Workers – support for multi-threading
 - ▶ And many more

HTML5

Taxonomy & Status (October 2014)

- Recommendation/Proposed
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Depreceted or inactive



HTML5 Canvas

- ▶ The HTML <canvas> element is used to draw graphics on a web page
- ▶ A canvas is a rectangular area on an HTML page
- ▶ By default, it has no border and no content
- ▶ You must use JavaScript to actually draw the graphics
- ▶ Canvas has several methods for drawing paths, boxes, circles, text, and adding images



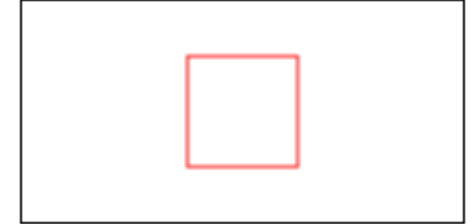
HTML5 Canvas Examples

▶ Drawing a rectangle

```
<canvas id="myCanvas" width="200" height="100"
style="border:1px solid black;"></canvas>

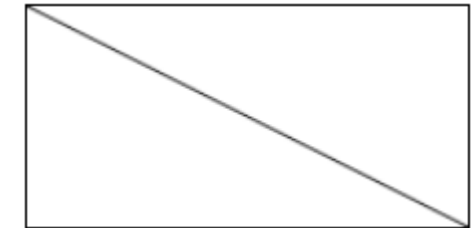
var canvas = document.getElementById("myCanvas");
var ctx = canvas.getContext("2d");

ctx.strokeStyle = "red";
ctx.strokeRect(75, 25, 50, 50);
```



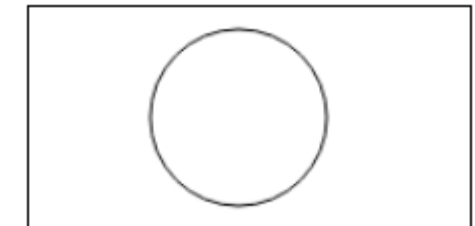
▶ Drawing a line

```
ctx.moveTo(0, 0);
ctx.lineTo(200, 100);
ctx.stroke();
```



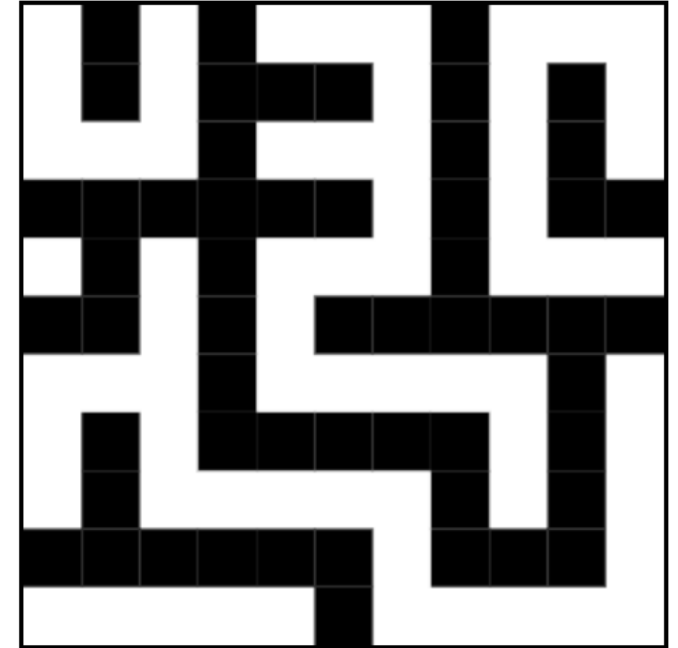
▶ Drawing a circle

```
ctx.beginPath();
ctx.arc(95, 50, 40, 0, 2 * Math.PI);
ctx.stroke();
```



Drawing a Maze on Canvas

```
<canvas id="mazeCanvas" width="300" height="300" style="border:2px black solid"></canvas>
<script>
  function drawMaze(maze) {
    var myCanvas = document.getElementById("mazeCanvas");
    var context = mazeCanvas.getContext("2d");
    var rows = maze.length;
    var cols = maze[0].length;
    var cellWidth = mazeCanvas.width / cols;
    var cellHeight = mazeCanvas.height / rows;
    for (var i = 0; i < rows; i++) {
      for (var j = 0; j < cols; j++) {
        if (maze[i][j] == 1) {
          context.fillRect(cellWidth * j, cellHeight * i,
cellWidth, cellHeight);
        }
      }
    }
  }
  maze = [[0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0],
          [0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0],
          ...
          [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]];
  drawMaze(maze);
</script>
```



HTML5 Local Storage

- ▶ Local storage allows web applications to store data locally in the user's browser
- ▶ The data is stored as a collection of key/value pairs
- ▶ Local storage is per origin (per domain and protocol)
 - ▶ All pages, from one origin, can store and access the same data
- ▶ Local storage provides two objects for storing data on the client:
 - ▶ `window.localStorage` - stores data with no expiration date
 - ▶ `window.sessionStorage` - stores data for one session (data is lost when the browser is closed)

```
// Store
localStorage.lastname = "Smith";

// Retrieve
document.getElementById("result").innerHTML = localStorage.lastname;
```

Bootstrap

Bootstrap

- ▶ Bootstrap is an open-source front-end web framework for designing websites
- ▶ Developed by Mark Otto and Jacob Thornton at Twitter
- ▶ Contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components
- ▶ Comes with several JS components that provide additional user interface elements such as dialog boxes, tooltips and carousels
- ▶ Supports responsive design
 - ▶ i.e., the layout of the web pages adjusts dynamically to the device used
- ▶ Includes a powerful flexible **grid system** for building layouts of all shapes and sizes

Getting Started

- ▶ Download Bootstrap from <http://getbootstrap.com/getting-started/#download>
- ▶ Once downloaded, unzip the compressed folder to see the structure of (the compiled) Bootstrap

```
bootstrap/  
├── css/  
│   ├── bootstrap.css  
│   ├── bootstrap.min.css  
│   ├── bootstrap-theme.css  
│   └── bootstrap-theme.min.css  
├── js/  
│   ├── bootstrap.js  
│   └── bootstrap.min.js  
└── fonts/  
    ├── glyphs-halflings-regular.eot  
    ├── glyphs-halflings-regular.svg  
    ├── glyphs-halflings-regular.ttf  
    └── glyphs-halflings-regular.woff
```

Basic Page Template

- ▶ Copy the HTML below to begin working with a minimal Bootstrap document
 - ▶ To use the JavaScript plugins jQuery is required

```
<!DOCTYPE html>
<html>
<head>
  <title>Bootstrap Demo</title>
  <meta charset="utf-8" />

  <link href="Styles/bootstrap.css" rel="stylesheet" />
</head>
<body>
  <h1>Hello, world!</h1>

  <script src="Scripts/jquery-3.2.1.js"></script>
  <script src="Scripts/bootstrap.js"></script>
</body>
</html>
```

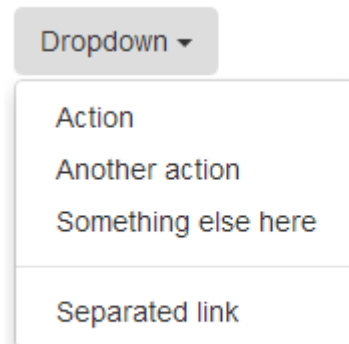
User Interface Components

► Buttons



```
<button class="btn btn-default">Default button</button>
<button class="btn btn-primary">Primary button</button>
<button class="btn btn-info">Info button</button>
<button class="btn btn-success">Success button</button>
<button class="btn btn-danger">Danger button</button>
```

► Dropdowns



```
<div class="dropdown">
  <button class="btn dropdown-toggle" type="button"
    id="dropdownMenu1" data-toggle="dropdown">
    Dropdown
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <li><a href="#">Action</a></li>
    <li><a href="#">Another action</a></li>
    <li><a href="#">Something else here</a></li>
    <li class="divider"></li>
    <li><a href="#">Separated link</a></li>
  </ul>
</div>
```

User Interface Components

► Lists

Link 1
Link 2
Link 3
Link 4
Link 5

```
<div class="list-group">
  <a href="#" class="list-group-item active">Link 1</a>
  <a href="#" class="list-group-item">Link 2</a>
  <a href="#" class="list-group-item">Link 3</a>
  <a href="#" class="list-group-item">Link 4</a>
  <a href="#" class="list-group-item">Link 5</a>
</div>
```

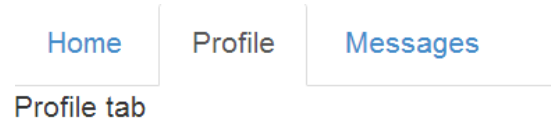
► Tables

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  <tr>
    <th>#</th>
    <th>First Name</th>
    <th>Last Name</th>
    <th>User Name</th>
  </tr>
  ...
</table>
```

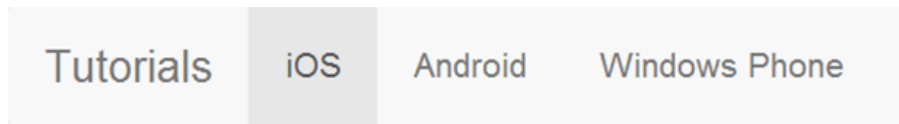
User Interface Components

► Tabs



```
<ul class="nav nav-tabs" id="myTab">
  <li class="active"><a href="#home">Home</a></li>
  <li><a href="#profile">Profile</a></li>
  <li><a href="#messages">Messages</a></li>
</ul>
<div class="tab-content">
  <div class="tab-pane active" id="home">Home tab</div>
  <div class="tab-pane" id="profile">Profile tab</div>
  <div class="tab-pane" id="messages">Messages tab</div>
</div>
```

► Navbars



```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
    <a class="navbar-brand" href="#">Tutorials</a>
  </div>
  <div>
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">iOS</a></li>
      <li><a href="#">Android</a></li>
      <li><a href="#">Windows Phone</a></li>
    </ul>
  </div>
</nav>
```

Grid System

- ▶ Bootstrap includes a powerful grid system for building layouts of all shapes and sizes
- ▶ It is based on a 12 column layout and has multiple tiers, one for each media query range
- ▶ Grid system rules:
 - ▶ Rows must be placed within a .container (fixed-width) or .container-fluid (full-width)
 - ▶ The class .row creates horizontal groups of columns
 - ▶ Grid columns are created by specifying the number of 12 available columns you wish to span
 - ▶ e.g., three equal columns would use three .col-sm-4
 - ▶ Move columns to the right using .col-sm-offset-*

```
<div class="container">  
  <div class="row">  
    <div class="col-*-*"></div>  
  </div>  
  <div class="row">  
    <div class="col-*-*"></div>  
    <div class="col-*-*"></div>  
    <div class="col-*-*"></div>  
  </div>  
  <div class="row">  
    ...  
  </div>  
</div>
```

Grid Classes

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-
# of columns	12			
Column width	Auto	60px	78px	95px

```

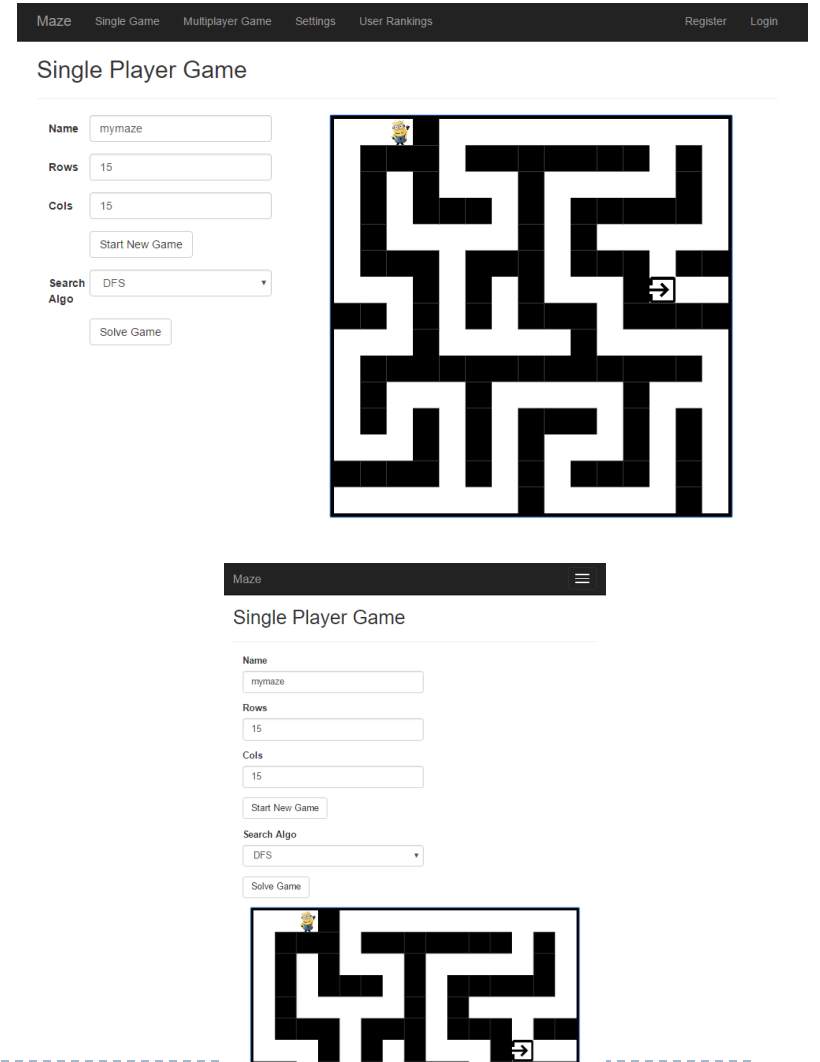
<div class="container">
  <div class="row">
    <div class="col-xs-6 col-md-3">
      <a href="#" class="thumbnail">
        
      </a>
    </div>
    <div class="col-xs-6 col-md-3">
      <a href="#" class="thumbnail">
        
      </a>
    </div>
    ...
  </div>
</div>

```



Grid System Example

```
<div class="container body-content">
  <h2>Single Player Game</h2><hr />
  <div class="col-sm-4 form-horizontal">
    <div class="form-group">
      <label for="mazeName" class="col-sm-2 control-label">Name</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="mazeName" />
      </div>
    </div>
    ...
  </div>
  <div class="col-sm-8 text-center">
    <canvas id="mazeCanvas" tabindex="1" class="maze-board" width="500"
height="500"></canvas>
  </div>
</div>
```




Bootstrap Templates

- ▶ You can find many more examples for Bootstrap templates on their web site (<http://www.getbootstrap.com>)

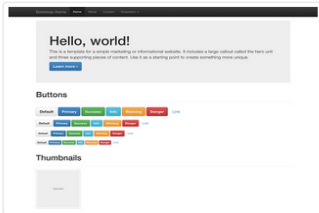
Examples

Build on the basic template above with Bootstrap's many components. See also [Customizing Bootstrap](#) for tips on maintaining your own Bootstrap variants.

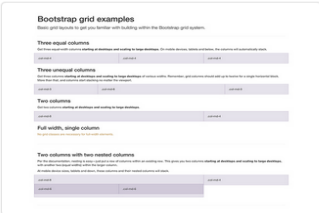
Using the framework



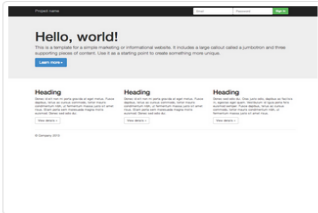
Starter template
Nothing but the basics: compiled CSS and JavaScript along with a container.



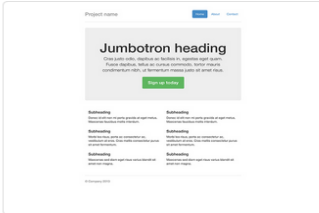
Bootstrap theme
Load the optional Bootstrap theme for a visually enhanced experience.



Grids
Multiple examples of grid layouts with all four tiers, nesting, and more.



Jumbotron



Narrow jumbotron
... a more custom page by narrowing