**OpenLayers Basics**

[OpenLayers](http://openlayers.org/)is a library for building mapping applications in a browser. This workshop covers the well established 2.x series, while development of a new 3.x series with a different API has begun. The library lets developers integrate data from a variety of sources, provides a friendly API, and results in engaging and responsive mapping applications.

**What this module covers**

In this module you will create a map, dissect your map to understand the parts, and get links to addi- tional learning resources.

**Creating a Map**

In OpenLayers, a map is a collection of layers and various controls for dealing with user interaction. A map is generated with three basic ingredients: *[markup](#_bookmark9)*, [*style declarations*](#_bookmark10), and *[initialization code](#_bookmark11)*.

**Working Example**

Let’s take a look at a fully working example of an OpenLayers map.

*<!DOCTYPE html>*

<**html**>

<**head**>

<**title**>My Map</**title**>

<**link** rel="stylesheet"href="openlayers/theme/default/style.css"type=

*˓→*"text/css">

<**style**>

**#map-id** {

width:512px; height:256px;

}

</**style**>

<**script** src="openlayers/lib/OpenLayers.js"></**script**>

</**head**>

<**body**>

<**h1**>My Map</**h1**>

<**div** id="map-id"></**div**>

<**script**>

**var** map= **new** OpenLayers.Map("map-id");

**var** imagery= **new** OpenLayers.Layer.WMS( "Global Imagery",

["http://maps.opengeo.org/geowebcache/service/wms"](http://maps.opengeo.org/geowebcache/service/wms),

{layers:"bluemarble"}

);

map.addLayer(imagery); map.zoomToMaxExtent();

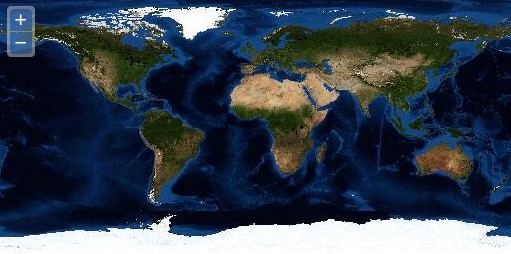
</**script**>

</**body**>

</**html**>

**Tasks**

1. Copy the text above into a new file called map.html, and save it in the root of the workshop folder.



Open the working map in your web browser:http://localhost:8082/ol\_workshop/map.html

Fig. 2.1: A working map of displaying imagery of the world.

**Note:** You can use *medit* (*Applications Accesories medit*) as your HTML editor for this workshop, it’s easy to use and has several nice features like syntax highlighting, line numbers, file explorer panel, etc.

**Dissecting Your Map**

As demonstrated in the [*previous section*](#_bookmark6), a map is generated by bringing together [*markup*](#_bookmark9), [*style declara-*](#_bookmark10)[*tions*](#_bookmark10), and [*initialization code*](#_bookmark11). We’ll look at each of these parts in a bit more detail.

**Map Markup**

The markup for the map in the [*previous example*](#_bookmark7) generates a single document element:

<**div** id="map-id"></**div**>

This <div> element will serve as the container for our map viewport. Here we use a <div> element, but the container for the viewport can be any block-level element.

In this case, we give the container an id attribute so we can reference it easily elsewhere.

**Map Style**

OpenLayers comes with a default stylesheet that specifies how map-related elements should be styled. We’ve explicitly included this stylesheet in the map.html page (<link rel="stylesheet" href="openlayers/theme/default/style.css" type="text/css">).

OpenLayers doesn’t make any guesses about the size of your map. Because of this, following the default stylesheet, we need to include at least one custom style declaration to give the map some room on the page.

<**link** rel="stylesheet"href="openlayers/theme/default/style.css"type="text/css">

<**style**>

**#map-id** {

width:512px; height:256px;

}

</**style**>

In this case, we’re using the map container’s id value as a selector, and we specify the width (512px) and the height (256px) for the map container.

The style declarations are directly included in the <head> of our document. In most cases, your map related style declarations will be a part of a larger website theme linked in external stylesheets.

**Note:** OpenLayers enforces zero margin and padding on the element that you use for the viewport container. If you want your map to be surrounded by some margin, wrap the viewport container in another element with margin or padding.

**Map Initialization**

The next step in generating your map is to include some initialization code. In our case, we have included a <script> element at the bottom of our document <body> to do the work:

<**script**>

**var** map= **new** OpenLayers.Map("map-id");

**var** imagery= **new** OpenLayers.Layer.WMS( "Global Imagery",

["http://maps.opengeo.org/geowebcache/service/wms"](http://maps.opengeo.org/geowebcache/service/wms),

{layers:"bluemarble"}

);

map.addLayer(imagery); map.zoomToMaxExtent();

</**script**>

**Note:** The order of these steps is important. Before our custom script can be executed, the OpenLay- ers library must be loaded. In our example, the OpenLayers library is loaded in the <head> of our document with <script src="openlayers/lib/OpenLayers.js"></script>.

Similarly, our custom map initialization code (above) cannot run until the document element that serves as the viewport container, in this case <div id="map-id"></div>, is ready. By including the initial- ization code at the end of the document <body>, we ensure that the library is loaded and the viewport container is ready before generating our map.

**var** map= **new** OpenLayers.Map("map-id");

Let’s look in more detail at what the map initialization script is doing. The first line of our script creates a new OpenLayers.Map object:

We use the viewport container’s id attribute value to tell the map constructor where to render the map. In this case, we pass the string value "map-id" to the map constructor. This syntax is a shortcut for convenience. We could be more explicit and provide a direct reference to the element (e.g. document. getElementById("map-id")).

The next several lines create a layer to be displayed in our map:

**var** imagery= **new** OpenLayers.Layer.WMS( "Global Imagery",

["http://maps.opengeo.org/geowebcache/service/wms"](http://maps.opengeo.org/geowebcache/service/wms),

{layers:"bluemarble"}

);

map.addLayer(imagery);

Don’t worry about the syntax here if this part is new to you. Layer creation will be covered in another module. The important part to understand is that our map view is a collection of layers. In order to see a map, we need to include at least one layer.

The final step is to set the geographical limits (xmin, ymin, xmax, ymax) of the map display. This *extent* specifies the minimum bounding rectangle of a map area. There are a number of ways to specify the initial extent. In our example, we use a simple request to zoom to the maximum extent. By default, the maximum extent is the world in geographic coordinates:

map.zoomToMaxExtent();

You’ve successfully dissected your first map! Next let’s [*learn more*](#_bookmark12) about developing with OpenLayers.

**OpenLayers Resources**

The OpenLayers library contains a wealth of functionality. Though the developers have worked hard to provide examples of that functionality and have organized the code in a way that allows other expe- rienced developers to find their way around, may users find it a challenge to get started from scratch.

**Learn by Example**

New users will most likely find diving into the OpenLayer’s example code and experimenting with the library’s possible functionality the most useful way to begin.

* <http://openlayers.org/dev/examples/>

**Browse the Documentation**

For further information on specific topics, browse the growing collection of OpenLayers documenta- tion.

* <http://docs.openlayers.org/>

**Find the API Reference**

After understanding the basic components that make-up and control a map, search the API reference documentation for details on method signatures and object properties.

* <http://dev.openlayers.org/apidocs/files/OpenLayers-js.html>

**Join the Community**

OpenLayers is supported and maintained by a community of developers and users like you. Whether you have questions to ask or code to contribute, you can get involved by signing up for one of the mailing lists and introducing yourself.

* Users list<http://lists.osgeo.org/mailman/listinfo/openlayers-users>
* Developers list<http://lists.osgeo.org/mailman/listinfo/openlayers-dev>