Rock Developer Quick-Start Guide

Subtitle The basics needed to start developing "stuff" for the Rock-ChMS Framework

Releaseinfo Version 0.02

Copyright 2011 Spark Development Network, LLC.

Abstract This guide is meant to be a starting point to quickly get new developers familiar with the Rock Framework. Once developers have mastered the basics outlined here they should proceed to the Rock Complete Developer Reference.

1 What is Rock ChMS?

Rock is an powerful application development framework which can be used to create a completely custom application or it can be used to develop additional functionality for the Rock Church Management System (ChMS). Rock is written using the latest technology including LINQ and the Entity Framework 4.0 but we'll leave that for another document so we don't put you to sleep. All you need to know is Rock was built to make development simple for you!

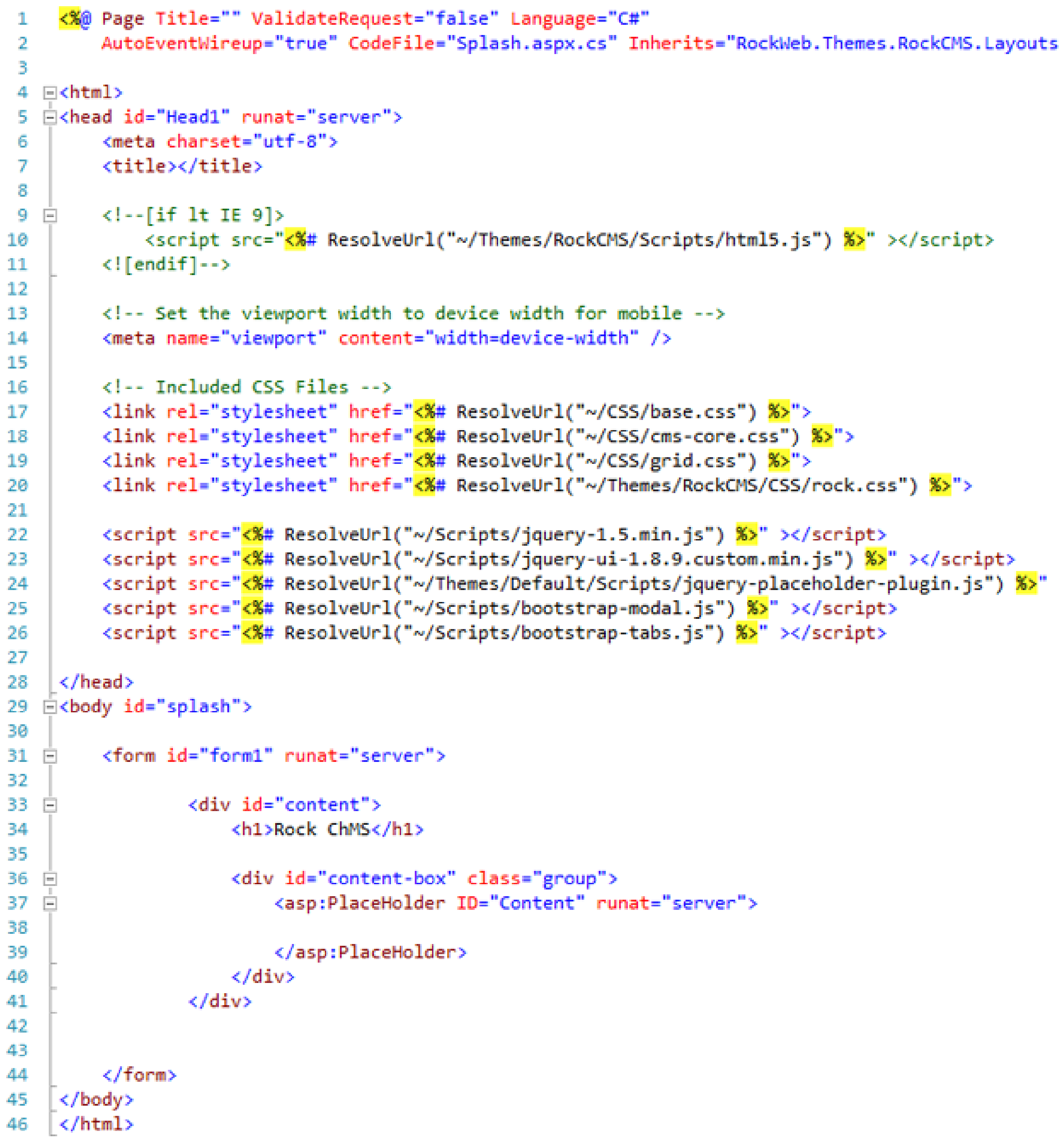
From a simplified perspective, Rock is primarily made up of two parts, *Theme Layouts* and *Blocks*. Themes have layouts that control the placement and definition of content areas called "Zones" and also include the pieces that define the styling. Blocks are the fundamental, functional code-parts that make up an application, widget, control, module -- or whatever you you call them.

When combined, Layouts and Blocks allow you to compose a powerful application. In fact, the ChMS parts of Rock are built using these very things. We like to say "Rock is built on Rock".

2 Theme Layouts

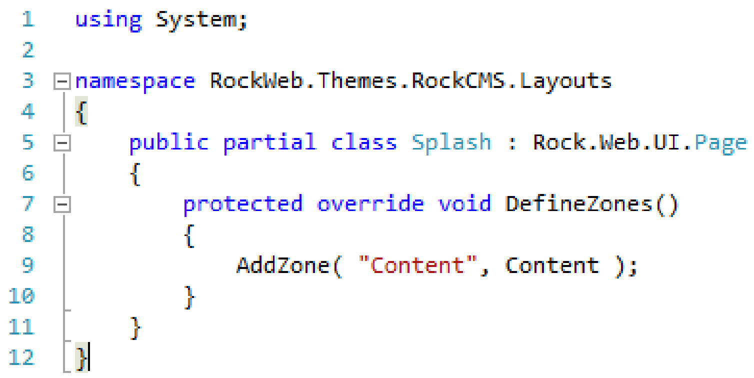
A Theme Layout is really just HTML. It's simply an ASP.NET Web Page (.aspx) that basically defines zero or more "zones" where Blocks can reside and will also refer to css, javascripts, and images. Theme Layouts can include just about anything you wish, *but they must inherit from the Rock.Web.UI.Page class* as shown in the following example.

An example Theme Layout, Splash.aspx, which has a single Zone



```
<%@ Page Title="" ValidateRequest="false" Language="C#"
    AutoEventWireup="true" CodeFile="Splash.aspx.cs" Inherits="RockWeb.Themes.RockCMS.Layouts

<html>
<head id="Head1" runat="server">
    <meta charset="utf-8">
    <title></title>

    <!--[if lt IE 9]>
        <script src="<%# ResolveUrl("~/Themes/RockCMS/Scripts/html5.js") %>" ></script>
    <![endif]-->

    <!-- Set the viewport width to device width for mobile -->
    <meta name="viewport" content="width=device-width" />

    <!-- Included CSS Files -->
    <link rel="stylesheet" href="<%# ResolveUrl("~/CSS/base.css") %>">
    <link rel="stylesheet" href="<%# ResolveUrl("~/CSS/cms-core.css") %>">
    <link rel="stylesheet" href="<%# ResolveUrl("~/CSS/grid.css") %>">
    <link rel="stylesheet" href="<%# ResolveUrl("~/Themes/RockCMS/CSS/rock.css") %>">

    <script src="<%# ResolveUrl("~/Scripts/jquery-1.5.min.js") %>" ></script>
    <script src="<%# ResolveUrl("~/Scripts/jquery-ui-1.8.9.custom.min.js") %>" ></script>
    <script src="<%# ResolveUrl("~/Themes/Default/Scripts/jquery-placeholder-plugin.js") %>"
    <script src="<%# ResolveUrl("~/Scripts/bootstrap-modal.js") %>" ></script>
    <script src="<%# ResolveUrl("~/Scripts/bootstrap-tabs.js") %>" ></script>

</head>
<body id="splash">

    <form id="form1" runat="server">

            <div id="content">
                <h1>Rock ChMS</h1>

                <div id="content-box" class="group">
                    <asp:PlaceHolder ID="Content" runat="server">

                    </asp:PlaceHolder>
                </div>
            </div>


    </form>
</body>
</html>
```

▽ *Code behind file (Splash.aspx.cs) for the Splash layout*

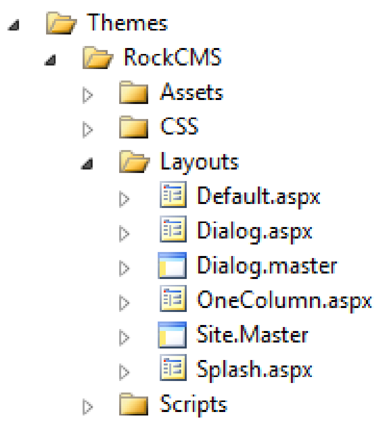


```
using System;

namespace RockWeb.Themes.RockCMS.Layouts
{
    public partial class Splash : Rock.Web.UI.Page
    {
        protected override void DefineZones()
        {
            AddZone( "Content", Content );
        }
    }
}
```

Notice how a single Zone named "Content" is added by defining a DefineZones() method which is overridden from the parent Rock.Web.UI.Page class. Any ASP control that can have child controls (such as the asp:PlaceHolder) can be used as a zone.

The recommended folder structure for any custom themes you create should be similar to the default *RockCMS* theme which has four layouts (Default, Dialog, OneColumn, and Splash) as shown below.

Sample folder structure for the RockCMS theme



```
Themes
  RockCMS
    Assets
    CSS
    Layouts
      Default.aspx
      Dialog.aspx
      Dialog.master
      OneColumn.aspx
      Site.Master
      Splash.aspx
    Scripts
```

We also recommend an Assets folder be used to store any fonts, icons and images your theme uses. Likewise, the CSS and Scripts folder would be used to store stylesheets and any JavaScripts (respectively).

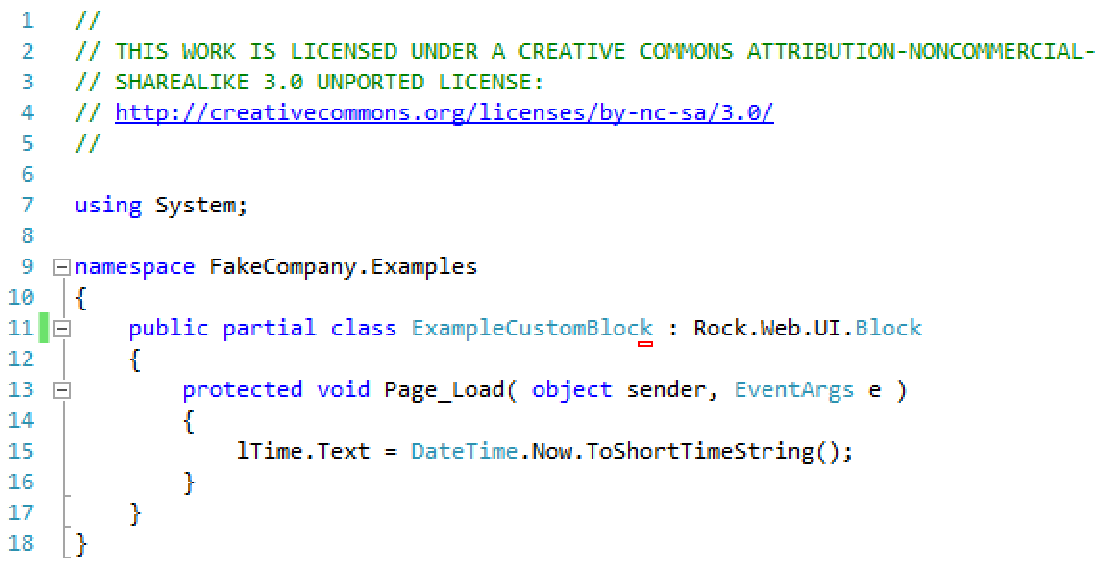
3 Blocks

Blocks are usually small pieces of HTML and code (your custom ASP.NET User Controls) and are the fundamental, functional "building blocks" that make up your plugin, widget, application or whatever you want to call it. They can include anything you want but they *must inherit from the Rock.Web.UI.Block class* as shown in the following example which is using a Code Behind file.

An example Block (ExampleCustomBlock.ascx)

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="ExampleCustomBlock.ascx.cs"
    Inherits="FakeCompany.Examples.ExampleCustomBlock" %>

<h2>An Example Block</h2>
The time is <asp:Literal ID="lTime" runat="server"></asp:Literal>
```

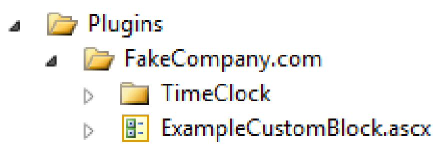
The code-behind for the example Block (ExampleCustomBlock.ascx.cs)



```
//
// THIS WORK IS LICENSED UNDER A CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL-
// SHAREALIKE 3.0 UNPORTED LICENSE:
// http://creativecommons.org/licenses/by-nc-sa/3.0/
//

using System;

namespace FakeCompany.Examples
{
    public partial class ExampleCustomBlock : Rock.Web.UI.Block
    {
        protected void Page_Load( object sender, EventArgs e )
        {
            lTime.Text = DateTime.Now.ToShortTimeString();
        }
    }
}
```

The standard location for all custom blocks is in the Plugins folder. We recommend you create your own folder under the Plugins folder to hold any custom blocks created by your team:

Location of Custom Blocks



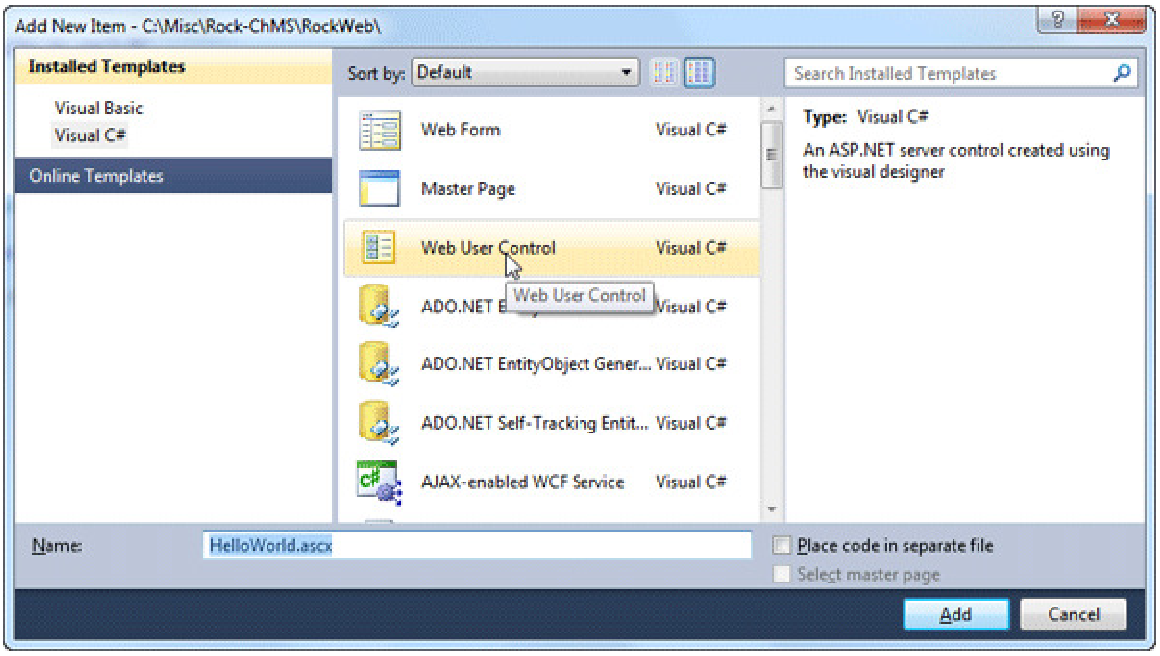
```
Plugins
  FakeCompany.com
    TimeClock
    ExampleCustomBlock.ascx
```

4 Hello World

Now let's walk through an complete example so you can see how easy it is to create a custom block and assemble a simple page. EDITOR NOTE: This should be turned into a video screen capture.

Create a "MyStuff" folder under the Plugins folder, then right click it an select "Add New Item". Next choose "Web User Control", uncheck the "Place code in separate file" checkbox and give it the name HelloWorld.ascx.

Choose Web User Control and name it HelloWorld.ascx


Add New Item - C:\Misc\Rock-ChMS\RockWeb\
Installed Templates
Visual Basic
Visual C#
Online Templates
Sort by: Default
Search Installed Templates
Web Form Visual C#
Master Page Visual C#
Web User Control Visual C#
ADO.NET E... Visual C#
Web User Control
ADO.NET EntityObject Gener... Visual C#
ADO.NET Self-Tracking Entit... Visual C#
AJAX-enabled WCF Service Visual C#
Type: Visual C#
An ASP.NET server control created using the visual designer
Name: HelloWorld.ascx
Place code in separate file
Select master page
Add
Cancel

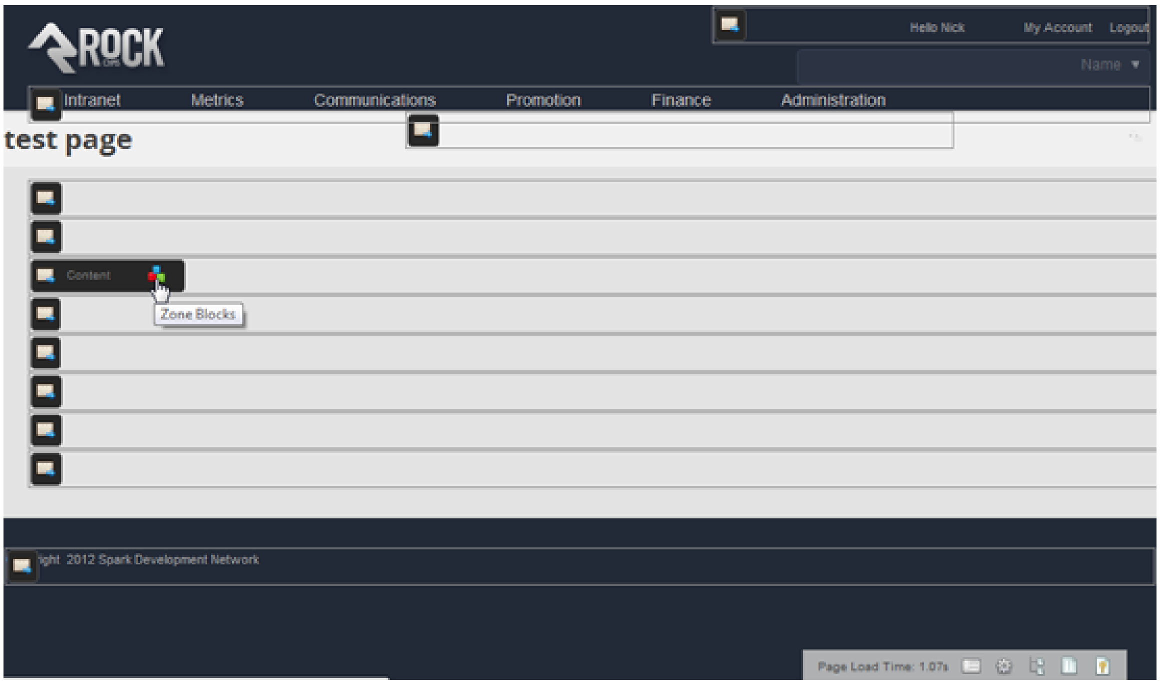
Add the *Inherits="Rock.Web.UI.Block"* string in the first Control line and type your Hello World as HTML as shown in the screenshot. *Note: If you're using code-behind file (*.ascx.cs) you'll need to change the inheritance in the *.cs file instead.*

The complete HelloWorld.ascx block

```
<%@ Control Language="C#" ClassName="HelloWorld" Inherits="Rock.Web.UI.Block" %>

<h2>Hello World</h2>
```

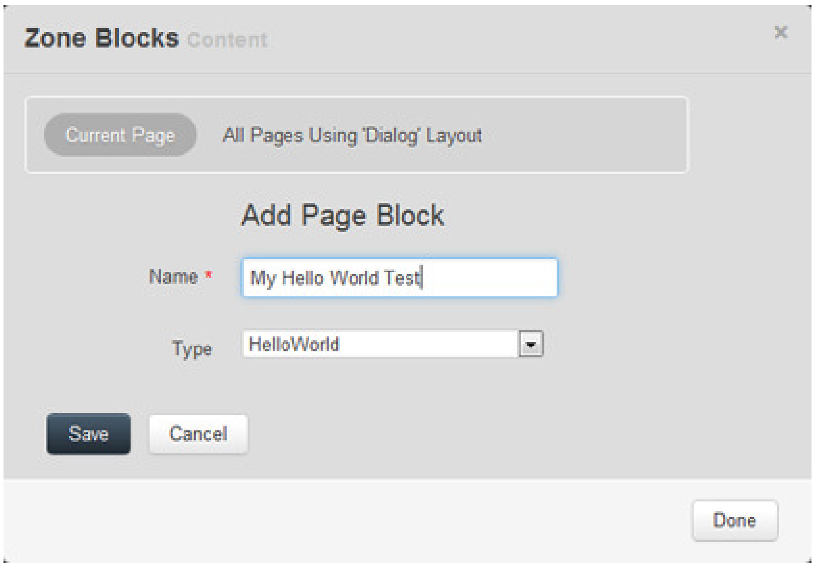
Now open Rock ChMS and login as an Administrator. Next, navigate to a test page and then choose the "Page Zones" icon from the toolbar in the bottom right corner. This is how you add Blocks to the Zones on a page. Select the *Content* zone's Zone Blocks as shown here:

Editing the zones on a page



Select the Add New button from the dialog box that appears. Doing this will search for all registered blocks and also will register any new blocks, including the new HelloWorld block you just created. Select your HelloWorld block Type and give it a name on this page.

Adding a block to a page zone

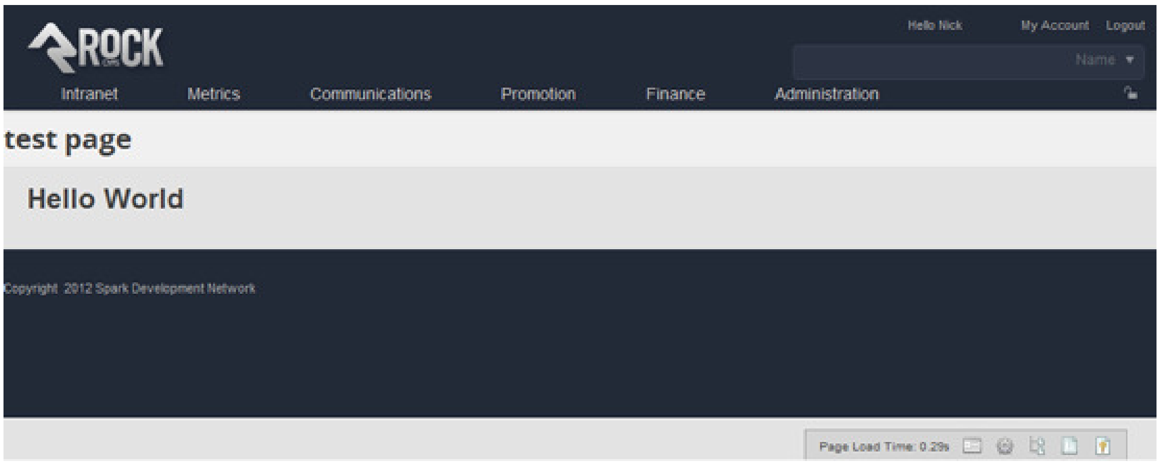


Name * My Hello World Test

Type HelloWorld

Once the page is refreshed, you will see an instance of your block on the page.

A test page showing an instance of your HelloWorld block


Hello Nick
My Account
Logout
Name
Intranet
Metrics
Communications
Promotion
Finance
Administration
test page
Hello World
Copyright 2012 Spark Development Network
Page Load Time: 0.29s

5 Pages

TBD