OpenSocial

Why is MySpace doing OpenSocial?

Reading this should give you a brief introduction to OpenSocial and how it fits into MySpace.

A Brief History of User Applications on MySpace

Let's start with an explanation of where we're coming from. MySpace has a very interesting history with applications written by users. You, gentle reader, may very well have participated in that history.

User coded, installable JavaScript applications have been on MySpace since it first launched. For those of you who were around since the beginning, you may recall that your user profile was a blank canvas on which you could paint with an unfettered pallette of HTML, JavaScript, and CSS. There were no constraints on what you could do, so long as a browser could render it.

MySpace was certainly not the first site to allow such things--there were many sites and services that allowed you to build your own "personal website" using any technology you desired. What set MySpace apart was that it was both personal site and social networking application: you could customize your site to your heart's content *and* you could interact with your friends in a variety of ways.

While unrestrained CSS and HTML provided users with limitless ways to make their profiles look a certain way, it was JavaScript that allowed them to really plug into the MySpace experience. After MySpace launched, users began building JavaScript widgets that did anything from customizing friends lists to sending MySpace Mail. And applications they coded were not limited to their own profiles. Through a little known technology known as "cut and paste", users could "install" applications they liked on their own profiles.

Where did all this functionality come from? While no specific XML/JSON api was provided, users quickly wrote and disseminated scripts that used JavaScript to screen scrape the existing MySpace markup (in order to gather data), and to emit the proper http values to manipulate the data they gathered.

Of course, a completely open MySpace was a utopic ideal. The exploitation began. As nefarious people began perceiving value in having lots of illegitimate friends, causing mischief, and/or making a profit through spam, they began writing applications that broke the rules. While a well thought out, law abiding "send me a message" app would send messages only at the request of the user, an app built by a spammer would send as many messages as the user's bandwidth would allow.

As spammers propagated through the site, MySpace began blacklisting certain types of JavaScript, HTML, and CSS. We tried very hard to keep as much JavaScript as possible, but slowly and surely illegitimate users hacked away at our filters until finally JavaScript was banned entirely. That left third party application developers with only one dyanmic alternative: Flash. Sites like YouTube saw their birth as widely disseminated Flash decorations for MySpace profiles. Unfortunately, by this time such applications were completely locked out of the MySpace data stream.

So--why is MySpace doing OpenSocial? Because every time we locked down a new JavaScript exploit we were sad, because we knew that legitimate application developers were getting hobbled as a result. For every ten spammers we blocked, we were blocking at least a few people trying to make a living by entertaining our users in a positive way. The OpenSocial platform gives us a chance to let MySpace users play again--this time in a safer, more structured, but at the same time more flexible way.

Enough history. Time for some code. By the time you're done with this document, you should be able to create a few sample OpenSocial apps on MySpace.

What is the MySpace Developer Platform?

The MySpace Developer Platform is a combination of services that provides third party application developers with hooks into MySpace's data and functionality. These services include:

- A suite of online tools for creating and publishing applications (and debugging them).set of RESTful APIs (provided in json, xml, and other formats as needed) that provide endpoints for browser-to-site and site-to-site interaction. These are implemented over the http protocol using a simple, intuitive uri scheme.
- A mechanism for your application to exchange data with your own site.
- A system for end users to find and install applications on their profiles.
- Security mechanisms for protecting end users' identity, as well as communications verification between MySpace servers and your site.

In short, the MySpace Developer Platform allows you to create stable, secure applications that (if you add the magic sauce) entertain MySpace users and allow them to explore the "social graph" in new and interesting ways. It also enables you to integrate your own site's functionality into MySpace. What do you get out of it? By writing an application that successfully entertains MySpace users (and believe us when we say they are fickle), you will gain fame, recognition, and traffic to your own site or service.

What is OpenSocial?

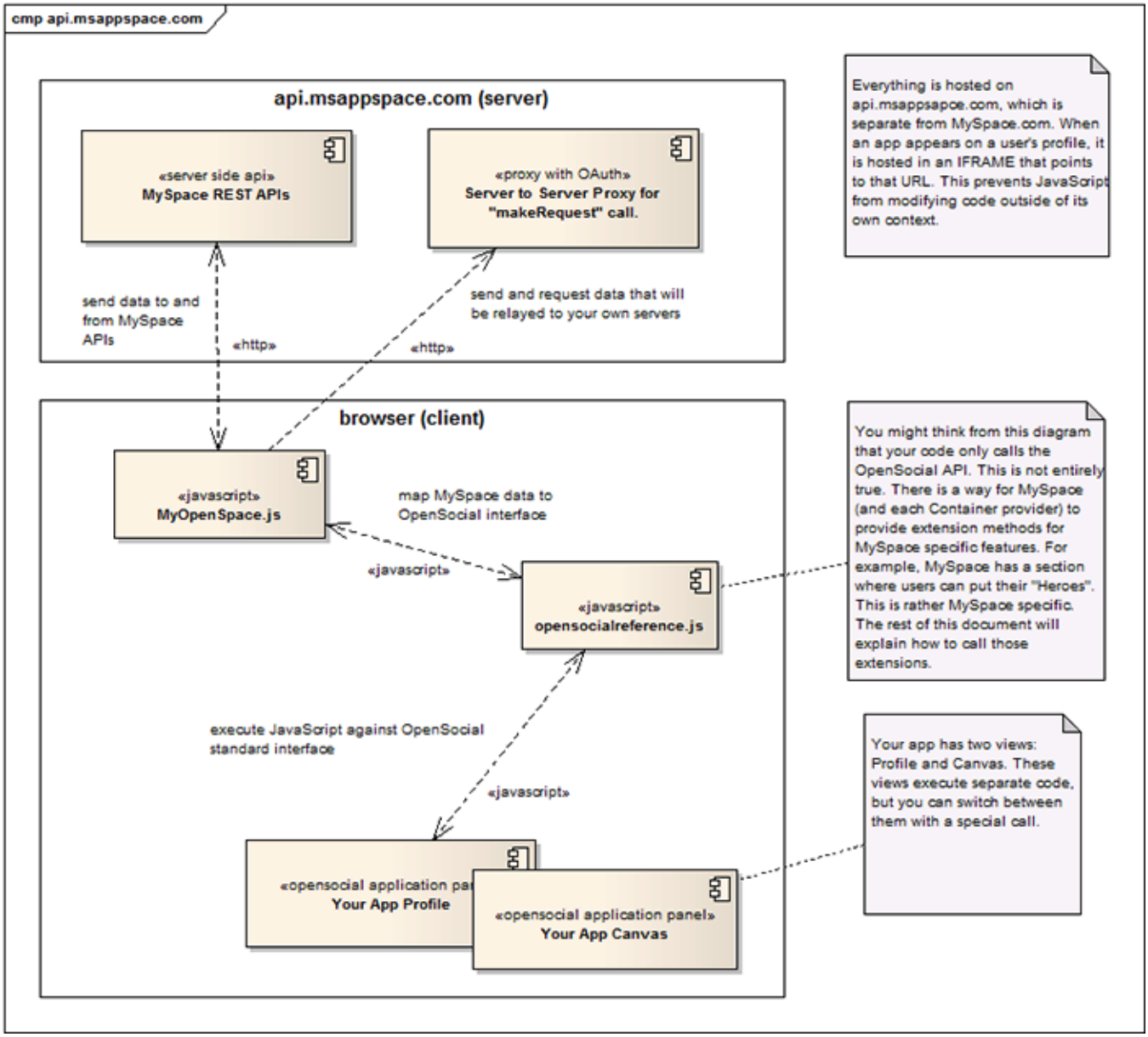
OpenSocial is a JavaScript API (Application Programming Interface) built in collaboration with Google and several other social networking sites. OpenSocial sites between your code and the MySpace Developer Platform.

How do OpenSocial and the MySpace Developer Platform fit together?

OpenSocial and the MySpace Developer Platform are a series of JavaScript and server-side components that work together to provide you with a standard interface to write your apps against.

Below is a rough picture of the communication between your app, MySpace, and OpenSocial. As you can see, OpenSocial lives completely on the client. There is no OpenSocial server and no communication between your app and some "OpenSocial" entity out there on the Internet.

figure 10 : Diagram of OpenSocial and MySpace Developer Platform Components



When you run your own app, you'll see that it's hosted in an IFrame. If you view the source around your code, you'll see a reference first to "opensocialreference.js", then to "MyOpenSpace.js". You should

study these files carefully, for they contain the meat of the OpenSocial implementation, as well as MySpace's extensions on it.

How do opensocialreference.js and MyOpenSpace.js communicate? MyOpenSpace.js is dependent on opensocialreference.js. OpenSocial defines several basic social networking objects, as well as implementations for how to get and update those objects. MyOpenSpace, meanwhile, maps those objects to MySpace server-side API calls. At times, it also extends the objects (more on that later). Your code will hit the opensocial namespace almost exclusively, but be aware that it is the MyOpenSpace code that is doing the dispatching and data mapping. For example, the following code is the official OpenSocial request for user data:

code sample 126 : Basic data request

```
    function init()
{
    var dataRequest = opensocial.newDataRequest();

    //Create a request for the owner's friends
    var friendRequest =
dataRequest.newFetchPeopleRequest(opensocial.DataRequest.Group.OWNER_FRIENDS)
;
    //Add the request for processing.
    dataRequest.add(friendRequest);
    //Send the request, passing in a callback.
    dataRequest.send(response);

}
```

Underneath the covers, that request issues an Ajax call to the MySpace RESTful APIs, which return the results in JSON format. MyOpenSpace.js then maps the resulting JSON format to the opensocial.Person object.

Stuff that isn't Implemented Yet

- **Navigating between Profile and Canvas Panels.** This does not currently work. When it does, you will call a function "requestNavigateTo()" to make your way between profile and canvas pages.
- **Requesting data from Third Party Servers.** This will be done in the upcoming "makeRequest" method.
- **Activities and App Data.** App Data is a way for your app to store data. Activities are ways for your app to notify users of events that have occurred.

Now onto the fun stuff!!

- A quickie hello-world type app
- Creating a working app
- MySpace specific extensions
- Working apps for your cutting and pasting pleasure