API (Application Programming Interface)

APIs define a set of rules in which you can interact with a particular application. For web developers, all of those Useful URL's we covered are specifically designed to be used by other developers or web pages. With a set of governing principles on how an API should be designed, developers can access information in very logical steps.

For example, let's say we have a huge database of Pokemon and we want to allow other developers to access this database to create cool Pokemon applications like a Pokedex. We can create a series of URLs that map to important information in a predictable manner. These developers will need to read the documentation of your API to learn how to properly interact with your URLs and get predictable results. This is exactly what PokeAPI did. Their documentation can be found here: http://pokeapi.co/.

Every API is Different

Even though every API is designed with a set of governing principles in mind, every API you interact with will be different. It is up to the developer who wrote the program to specify the protocols needed to interact with that API. Some APIs will let you access information as long as you ask for it in the URL. Some APIs will give you better information if you submit more data through a form. Some APIs will only allow you to request a certain URL a limited number of times per day. Some APIs will require you to sign up on their website to be provided an authenticity token to pass with every API request you make.

APIs are usually web URLs where you can pass information either in the URL or by posting a form to the URL and the server returns JSON (or other similar data formats). You will create your own API in later chapters but in essence, that is all you need to have to build an API: if a URL takes some data and outputs JSON (or other similar formats), you have created an API. Take a quick look at some of the APIs listed below.

. Open Weather Map: http://openweathermap.org/api
. GitHub API: https://developer.github.com/v3/
. Google Maps Directions API: https://developers.google.com/maps/documentation/directions/
. Twitter API: https://dev.twitter.com/rest/public
. Flickr API: https://www.flickr.com/services/api/

Although you have to interact with each API differently, notice how they are all collections of URLs that you can interact with as long as you follow the protocols that the developer has specified in the documentation.

What's the Big Deal?

- **Speed (Developers):** Instead of creating everything from scratch, you can utilize APIs of other web services to build some cool apps more easily. There are lots of useful APIs out there.
- **Cross development (Managers/Developers):** APIs allow experts/developers of different languages (PHP, Ruby, Python, etc.) to work together efficiently. For example, let's say you had a team of Java developers who built an app in Java and you had another team of PHP developers who built an app in PHP. Say you had another team of JavaScript developers who built their app in Node. What could you do if you wanted these three apps to work together (send data back and

forth)? You could either have everyone migrate to just one language (not recommended) or just have three teams build APIs (URLs where you can submit/retrieve information) so that these three apps can work together to communicate information back and forth!

- **Wider reach of the audience (Marketers):** APIs allows marketers to market their product/services to a wider audience of developers. Then, these developers can build applications that use the API to reach an even wider audience of consumers and developers. For companies like Twitter, Facebook, Flickr, and Google, they wanted to reach out to a wider audience so they made it easy for developers to use their services to build their brand even more.