Understanding Infusion Components

One of the core concepts in Infusion is that of "components." Infusion components are JavaScript objects designed to modularize functionality. In an Infusion application, just about everything is a component. The main application is itself a component that coordinates communication between subcomponents, which themselves might have subcomponents. If you're accustomed to developing with object oriented languages like Java, this will be familiar to you, but if your experience is primarily with JavaScript, this might be new – most JavaScript frameworks don't offer this level of support for object orientation.

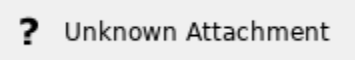
What does this mean for you? It depends what you're working on. If you're creating a single UI-widget-type-thing, your widget will likely be a component and may have several subcomponents. (The components in the infusion Component Library are created this way).

If you're creating an entire web application, your application would be implemented as a component that coordinates interactions between other components that handle the different parts of your application.

Examples

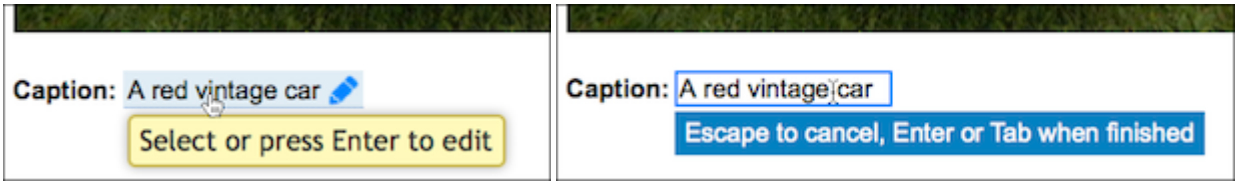
To help understand how a widget or application might be designed using components, consider some of the components in the Infusion Component Library:

Progress



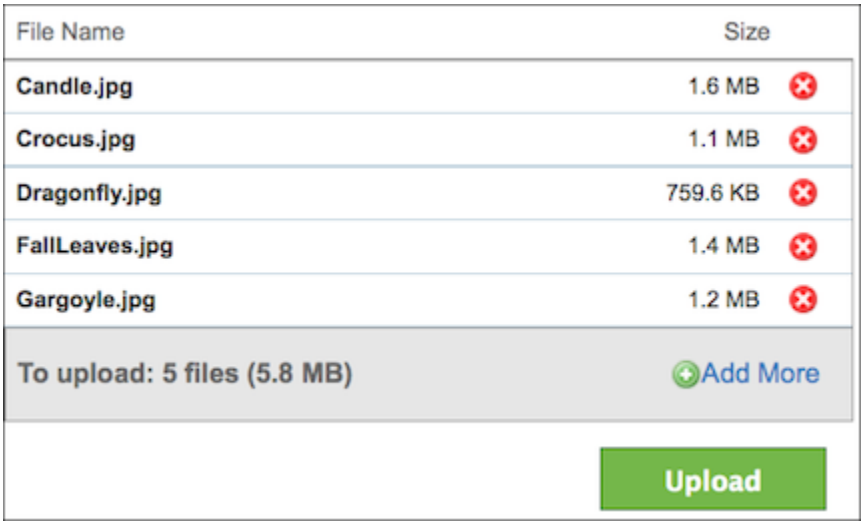
The Infusion Progress component is single component with no subcomponents. It has a number of UI elements that work together and are updated programmatically to show the progress of some activity. It has a pretty simple purpose and function, one that doesn't make much sense to try to chunk up into multiple components.

Inline Edit



The Inline Edit component allows user to edit text in place, without switching to a new screen, by simply switching into an in-place edit mode. The view mode is implemented one way, with certain functionality (i.e. a tooltip, an affordance to edit), and the edit mode is implemented differently: it's an edit field. Conceptually, these two modes are rather different, and so they're implemented as two separate subcomponents of the main Inline Edit component.

Uploader



The Uploader allows users to add several files to a queue and then upload them all at once. It is actually made up of several subcomponents: It has the file queue view, which displays the files currently in the queue; it has a total progress bar at the bottom. In turn, the file queue view component has its own subcomponents.

What Does A Component Look Like?

A component is a regular JavaScript object that has certain characteristics. The most simple components have a `typeName` and an `id`, but typical components will have more:

Most will have:

- a 'creator function'
    - the function that implementors invoke, which returns the component object itself
- configuration options
    - various values that control the operation of the component, which can be overridden by implementors to customize the component
- public functions

Depending on what the component is for, some will include infrastructure to support

- events
- a model
- a view
- a renderer

While you are free to manually write code to create your components (i.e. by writing a creator function, etc), the Framework provides supports for automatically creating components of various types, and we recommend that you use these supports.

Next: Understanding Component Options And Their Defaults