Positron FAQ

What is Positron?

Positron is a framework to help markup authors to easily write web applications. Functionality previously limited to manually written Javascript is made available to authors via extensible annotations to HTML.

What is Positron's ancestry?

Positron is effectively version 2 of Particle Programmatica's Neutrino framework. Neutrino was used to author an application for Cirque du Soleil website (movikantirevo.com) along with many others.

Is Positron just like Web Components and Angular, etc?

The web is moving towards annotating HTML for behaviour, so there will naturally be commonalities and differences between frameworks as this area is explored prior to standardisation.

Positron's target demographic is HTML authors, so it doesn't allow for embedded Javascript within markup. However, Positron offers a great deal of flexibility for attaching Javascript behaviour to markup elements such as tags and attributes, which can then be referenced without code.

Positron also provides a highly extensible markup-based shorthand for connecting events to actions, obviating boilerplate Javascript.

Why do some examples fail with Ajax or JSON errors?

The examples assume you're using a browser which allows loading resources off the filesystem using Ajax (providing the application was initially loaded off the filesystem). Currently this is limited to Safari with "Disable Local File Restrictions" turned on. If you're seeing these errors, switch to Safari for running the examples, or host them on a web server.

Which browsers are supported?

Positron depends on CSS3 animations, in theory any browser with good CSS3 support should work.

Why does Positron use CSS3 animations for transitions?

Transitions may seem like a better conceptual fit, but animations are more flexible and don’t have some of the limitations. Positron would support both, but sadly browser issues prevent it from doing so.

When should I use a Page, and when should I use a View?

A Page represents a bookmarkable application state - it has a mission, if you like. A View is simply a potentially reusable component which manages a section of screen, contextualised by its parent Page. (This is a short and trivial answer to a complex question.)

Why does Positron use CSS3 animations for transitions?

Transitions may seem like a better conceptual fit, but animations are more flexible and don’t have some of the limitations. Positron would support both, but sadly browser issues prevent it from doing so.

Why do Views or Pages get stuck in transitional states?

In order for a View or Page to transition to the visible or invisible state, the animation for the transition CSS class must successfully complete.

Situations which prevent animations from completing include:
a missing class associated with the transition class name used
an animation name which does not reference a set of key frames
the CSS definition is browser-neutral but is loaded with an href instead of nu- href attribute on the inclusion tag

How do I set the initial Page of the application?

The initial Page can be set by adding a “nu-start-page” attribute to <body>, which contains the key of the Page to load. Also gApplication.setPage() can be called any time after the call to gApplication.start() in main().

The initial Page can also be set on a request by request basis by specifying its key after the hash in the URL.

Why are action parameters not being passed?

A common mistake in action encoding is to exclude the second semicolon when expanding context variable values in action parameter attributes. The first semicolon is required for the context variable expansion; the second is required to delimit the parameter value.

```
<div
  pos-action=”showview: detail”
  pos-action-params=”record_id: $record.id;;”
  >
  Click here to show detail for record ID $record.id;
</div>
```