What's New in Infusion 2.0

Pirate Pad

Community Meeting: What's New in Infusion 2.0?

- http://docs.fluidproject.org/infusion/development/APIChangesFrom1_5To2_0.html
- Removal of lots of old features
  - Manual lifecycle points finalInit, postInit, etc.
  - Obsolete syntax for arguments, options, etc.
  - Removal of "autoInit"
  - Removal of the old model component hierarchy and "old ChangeApplier" implementation
- Context Awareness - and things it relies on:
  - Global instantiator
    - Every Infusion component, regardless of how it is instantiated, ends up in a single-rooted tree of components
    - This enables use of modern IoC features such as model relay and declarative event binding
    - Enables use of the root distributeOptions context "/"
    - This also enables the removal of "demands blocks"
    - Useful debugging tip: Watch "fluid.globalInstantiator" in your JS debugging tools to see the structure of your application and its tree.
- Much faster invokers and boiled listeners (c. 60x faster)
  - Removal of the old "fast/dynamic" invoker distinction
- View oriented IoC debugging tools
  - FluidViewDebugging.js - include this at the head of your HTML file to get access to the "IoC inspector"

Questions:

- When to use context awareness vs some other framework features.
  - http://docs.fluidproject.org/infusion/development/ContextAwareness.html#when-and-how-to-apply-fluid-contextaware-
  - Hiearchy of adaptation -
    - Simplest: Simply send extra constructor arguments to the component when it is made - either as function arguments or subcomponent arguments
    - Middling: Use a distributeOptions block to target the component from elsewhere
    - Most heavyweight: Make it "contextAware" and so respond to distributions from multiple sources in an organised and arbitrated way
  - In relatively straightforward cases, you can just write plain options distributions
  - Perhaps in conjunction with the so-called "file inclusion polymorphism"
    - That is, that the person who is aware of the contextual requirement can arrange to include different .js files into the system in order to represent that fact
    - Then there is no need for contextAwareness "adaptation" broadcasting components since you can simply write the distributeOptions directives in the files which are conditionally included
- How to add context awareness to a component that didn't previously support it. E.g. for testing purposes. (maybe i want to swap the TTS component for the mock in the test).
  - Simply add the "fluid.contextAware" grade to its grade list
- Is there a reason for "grades at the right-hand end of the gradeNames list now take priority over those at the left".
  - It is simple common sense :)
- Do model listeners support namespaces? such as having a "namespace" option at defining a model listener?

- No, they don't - see https://issues.fluidproject.org/browse/FLUID-5695