

Upcoming Store Developments

Introduction

- Describe current and upcoming work on Store
- Some in 7.5
- Some in preview or contributed
- Some still conceptual

Topics

- Shadow Compilation
- Merge Tool
- Glorp as a Store back-end
- Projects and Streams
- Internationalization

Shadow Compiling

Motivation

➔ Partial Code Loading

- ↪ Errors in the code being loaded
- ↪ Unloadable definitions
- ↪ Loading new versions of code involved in loading
- ↪ File-in ordering issues
 - » Add instance variable to superclass
 - » Remove instance variable from subclass

Normal Loading

➔ File-in

- ↪ Errors stop loading
- ↪ Code remains in image

➔ Parcels

- ↪ Code loaded into data structures
- ↪ Installation happens after all code read
- ↪ On error, image remains stable
- ↪ Supports uninstalled code

Store Loading

➔ Source

- ↪ Like file-in
- ↪ But handles unloadable definitions
- ↪ These can be applied later
- ↪ Still fails on errors

➔ Binary

- ↪ Like parcels
- ↪ Doesn't handle unloadable definitions

Shadow Compilation

- Like file-in, compiles source
- Like parcels, compiled elsewhere
- Create a separate set of “shadow” namespaces

Procedure

- Create a mirror of the namespace hierarchy
- Compile the code into those namespaces
 - ↳ Make a “copy” of the class for extensions
 - ↳ Interesting variable resolution issues
- Validate the code
- Install into the system

User's Point of View

- Everything works exactly like it did
- Loading is more robust
- “Atomic” loading

Status

- ➔ Supporting code included with 7.4.1
 - ↪ Not turned on
 - ↪ Some serious bugs remaining
- ➔ Included in 7.5
 - ↪ Probably not turned on by default
 - ↪ Setting exposed
 - ↪ Essentially preview
 - ↪ `Store.Bundle useShadowLoader: true.`

Merge Tool

Merge Tool Updates

- Significant Update for 7.5
- User interface
- Better indications of unusual cases
- But, doesn't change performance

User interface

- ➔ Hierarchical view
 - ↔ Can change between different views
 - » Method list
 - » By component/class
- ➔ Component-level operations
- ➔ Menus/Toolbar

Visibility

- ➔ Hide/show
 - ↳ Unresolved
 - ↳ Image-only mods
 - ↳ Conflicts
 - ↳ Full names
- ➔ Zoom text views

Semantics

- ➔ Better detection of moves
- ➔ Example
 - ↪ Net.URI moved to OS.URI
 - ↪ Backward-compatibility stub
 - ↪ WebToolkit extends Net.URI
- ➔ “(Possible move, see also OS.URI)”

StoreForGlorp

StoreForGlorp

- ➔ Glorp back-end for Store
 - ↪ Adds a domain model for Store database entities
 - ↪ Better performance
 - ↪ Richer queries
 - ↪ Support schema modifications
 - » Version migrations
 - » Dialect portability
 - ↪ Various Utilities
 - ↪ In preview
 - ↪ In use by various team members and others
 - » Niall Ross talk

StoreForGlorp Utilities/Features

- Replication/Auto-replication
- Publish
- Load
- Compare
- Merge
- Repository Crawler
- Browse Class versions
- Store workbook
- Auto-reconcile

Projects and Streams

Projects and Streams

- ➔ Address “configuration management” issues in Store
 - ↪ Lots of different meanings
- ➔ Early stage
- ➔ Design-level work
- ➔ Interested in feedback

Projects

- Higher-level code organization
- Tools aspect
 - ↳ Define a working set
 - ↳ Not going to talk about
- Code management aspect
 - ↳ Store
 - ↳ Deployment

Projects in Store

- Multiple independent components
- Can be large
 - ↳ Not all loadable in a single image
 - ↳ Too big
 - ↳ Incompatible
 - ↳ Conditionally loaded

Goals with Projects

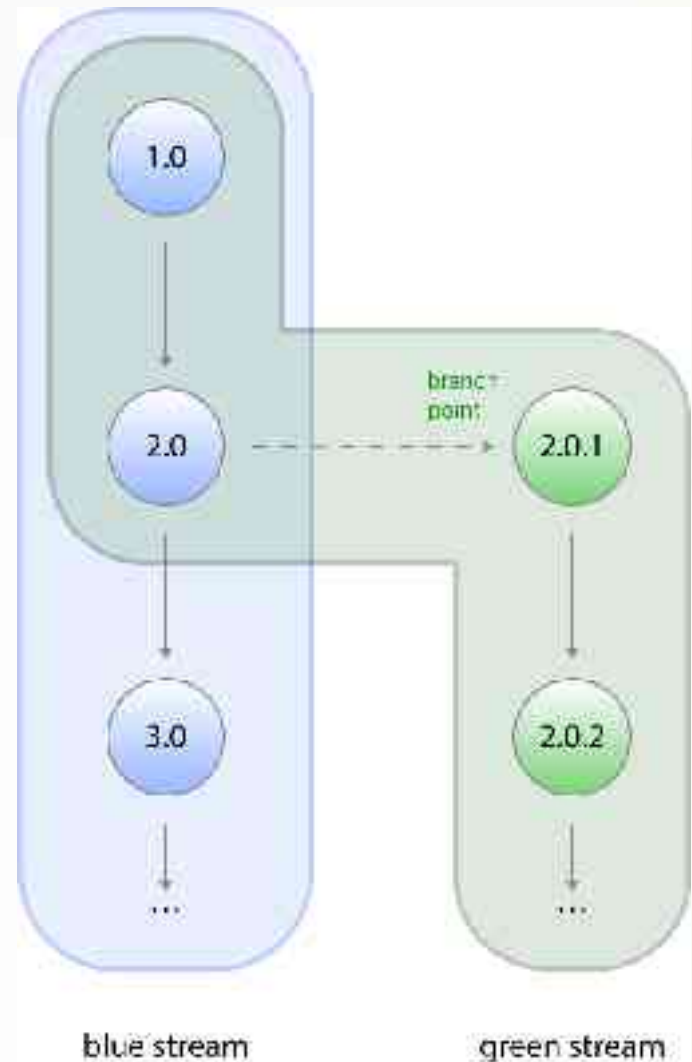
- ➔ Version control of larger-scale components
- ➔ Partially loadable
- ➔ Deployment-aware
- ➔ Aware of non-Smalltalk artifacts
- ➔ Allow people to work at any level of the component hierarchy
- ➔ Synchronizing work more easily
- ➔ Explicit branches

Scale

- ➔ Be able to version “VisualWorks”
 - ↳ Comparison to Bundles
- ➔ Specify loading
 - ↳ Manually
 - ↳ Conditional expressions
- ➔ Publish without everything loaded
- ➔ Sync up without everything loaded

Aside: Streams

- A series of versions
- Done now with naming conventions



Streams Enable

- ➔ Explicit notion of branching
- ➔ Explicit notion of syncing
 - ↪ Comparison to ENVY open editions
 - » History
 - » No close/open-new disruption
 - ↪ Comparison to “tags”
 - » Objects
 - » Version history
- ➔ Enables more automated merging

Work at Multiple Levels

- We have a project P, containing sub-projects Q, R, S. Each sub-project contains multiple packages/bundles.
- Branch P to work on a specific feature.
- Developers should be able to
 - ↳ Publish version of P
 - ↳ Publish version of Q,R, or S
 - ↳ Publish version of particular package.
- Sync up, whichever level we published at

Projects Summary

- Streams of components at all levels
- Model of syncing and merging
- Larger-level than can be handled with bundles
- Early prototype stage
- Superficial overview

Questions?