Glimmer

Creation Review

Outline

- Introduction
- Background and Goal
- Packaging and Deployment
- Project Scope
- Mentors
- Initial Committers
- Interested Parties

Introduction

- Glimmer is a JRuby DSL that enables easy and efficient authoring of user-interfaces using the robust platform-independent Eclipse SWT library.
- Glimmer comes with built-in data-binding support to greatly facilitate synchronizing UI with domain models.

Example

- The next pages show an example of a program written in SWT, JFace, and Glimmer, to clearly illustrate the benefits of Glimmer's syntax.

SWT Java Example

```
public static void main(String[] args) {
    Display display = Display.getDefault();
    Shell shell = new Shell(display);
    shell.setText("Example");
    shell.setLayout(new FillLayout());
    Composite composite = new Composite(shell, SWT.NONE);
    composite.setLayout(new GridLayout());
    Label label = new Label(composite, SWT.NONE);
    label.setText("Hello World!");
    shell.pack();
    shell.open();
    while (!display.isDisposed()) {
        if (!display.readAndDispatch()) {
            display.sleep();
        }
    }
    display.dispose();
}
```

JFace Java Example

```
protected JFaceTest(Shell parentShell) {
	super(parentShell);
	setBlockOnOpen(true);
	open();
}
protected Control createContents(Composite parent) {
	getShell().setText("Example");
	getShell().setLayout(new FillLayout());
	Composite composite = new Composite(parent, SWT.NONE);
	composite.setLayout(new GridLayout());
	Label label = new Label(composite, SWT.NONE);
	label.setText("Hello World!");
	return composite;
}
public static void main(String[] args) {
	new JFaceTest(null);
}
```

Glimmer Ruby Example

```
shell {
	text "Example"
	composite {
		label { text "Hello World!" }
	}
}.open
```

Background and Goals

- Ruby is a dynamically-typed object-oriented language
- Provides great productivity gains due to:
  - Powerful expressive syntax
  - Dynamic nature
- Proven by the Ruby on Rails framework for web development.

Background and Goals

- However, it currently lacks a robust platform-independent framework for building desktop applications.
- Given that Java libraries can now be utilized in Ruby code through JRuby, Eclipse technologies, such as SWT, JFace, and RCP can help fill the gap of desktop application development with Ruby.

Background and Goals

- The goal of the Glimmer project is to create a JRuby framework on top of Eclipse technologies to enable easy and efficient authoring of desktop applications by taking advantage of the Ruby language.

Packaging and Deployment

- As Glimmer solidifies its DSL to write SWT and RCP applications, the next challenge is to figure out how to elegantly integrate it as an Eclipse plug-in.
- Additionally, we will consider packaging Glimmer (or parts of it) as a RubyGem to make it easy to consume by the Ruby community.

Project Scope

- The scope of the project is simply to make it possible to use a JRuby framework to author Eclipse-based applications.
- Future work may involve enhancements to meet new use-cases or take advantage of other Eclipse technologies like the Eclipse Modeling Framework and other Ruby technologies like Rails.

Project Scope

- While Glimmer's original goal is to enable desktop application development with JRuby, that does not preclude it from growing to cover other areas, such as web development.
- For example, when SWT starts supporting Ajax widgets, Glimmer may be enhanced to further simplify web development (e.g. provide a Glimmer Rails plug-in.)

Mentors

- PDE - Chris Aniszczyk [zx@us.ibm.com]
- EMF - Ed Merks [merks@ca.ibm.com]

Initial Comitters and Bios

- Annas "Andy" Maleh (Obtiva)
  - Founded Glimmer project and contributed majority of source code and ideas. Experienced in RCP and Ruby on Rails development.
- Nick Malnick (Obtiva)
  - Contributed to Glimmer by pair-programming on listener support and the Tic Tac Toe sample game, researching integration with RCP, and researching better ways to do unit-testing. Experienced in RCP and Ruby on Rails development.

Initial Comitters and Bios

- Dave Hoover (Obtiva)
  - Contributed by encouraging and guiding the test-driven development process and helping with the creation of a RubyGem for the project. Experienced in Ruby and Rails development. Leads Obtiva's craftsmanship studio.
- Kevin P. Taylor (Obtiva)
  - Contributed by testing Glimmer on the Mac. Will be a future resource for verifying Mac functionality. Experienced in RCP and Ruby on Rails development. President of Obtiva Corp.

Interested Parties

- PDE – Chris Aniszczyk [zx@us.ibm.com]
- EMF – Ed Merks [merks@ca.ibm.com]
- Obtiva Corp. [obtiva.com]
- Sopera GmbH [sopera.de]
- Gerald Preissler [gerald.preissler@sopera.de]
- Ketan Padegaonkar [kpadegao@thoughtworks.com]
- Casey Marshall [casey.marshall@gmail.com]