Themes

Studio 3 uses a new "theming" system to define the coloring of the IDE across the various editors and views. We tried to follow the convention of token names used by TextMate in our coloring so that porting existing themes over would be relatively easy.

Scopes

Scopes are similar to CSS selectors in that they describe a pattern which is matched against the current location of the cursor in the document. Multiple scopes may apply simultaneously at different levels of specificity. As an example, if we take the following piece of CSS:

```
body {
    background: #000000;
}
```

The entire snippet in a CSS file will have the scope 'source.css'. Individual tokens will have the following scopes applied:

Token	Full Scope	Theme Scope Selector Rule that matches ("Aptana Studio" Theme)
body	source.css entity.name.tag.css	entity.name.tag.css
{}	source.css punctuation.section.property-list.css	punctuation.section
background	source.css support.type.property-name.css	support.type.property-name.css
:	source.css punctuation.separator.key-value.css	source
#000000	source.css constant.other.color.rgb-value.css	constant.other.color.rgb-value.css
;	source.css punctuation.terminator.rule.css	source

You can double-check this yourself by placing the cursor at a particular location, and choosing "Commands > Bundle Development > Show Scope". You will see each of the pieces of text above has "source.css" and the specific scope in question applied.

How does this relate to theming? If you open up Preferences > Aptana > Themes, you will see a list of scopes. When applying colors, it will choose the scope closest in match to the one above by following a prefix match, i.e. "punctuation.section.blah.blah" matches "punctuation.section" for which a color is defined.

Many of the scopes defined in themes are relatively general across languages. You can add scopes if you like to specifically modify things to your taste. Some examples:

Desired Effect	Action
Colorize semicolons in CSS	Add scope 'punctuation.terminator.rule.css' and choose colors
Colorize the background of CSS source	Add scope 'source.css' and choose colors
Colorize the background of CSS source, but only when embedded in any language	Add scope 'source.css.embedded' and choose colors.
Colorize the background of CSS source, but only when embedded inside an HTML document (i.e. a **style** tag)	Add scope 'text.html source.css' and choose colors.

There is a longer discussion about creating scopes here: http://manual.macromates.com/en/scope_selectors.html

Token names

A good starting point for token naming conventions is the TextMate documentation on the subject: http://manual.macromates.com/en/language_grammars#naming_conventions

Generally speaking tokens follow a convention of *category.sub-category.language-extension*, i.e. "comment.line.js"

See Current Theme Scopes for a complete list of currently available scopes.

Themes in Rubles

Rubles may contribute themes. Simply use a hash from token name to a string containing the fg, bg(optional) and font styles (optional). Place the theme addition in a file that will get loaded when the ruble does (i.e. bundle.rb). The overall theme color keys of foreground, background, selection, caret, and lineHighlight are expected to be present as keys; as is the "name" key for the theme name. The rest of the keys are interpreted as token names (if using symbols, "_" will be changed to "." under the hood for you). Here's an example:

```
require 'ruble/theme'

Ruble::Theme.add({
  :name => 'example',
  :foreground => '#ffffff',
  :background =>  '#000000',
  :selection => '#999999',
  :caret => '#cccccc',
  :lineHighlight => '#cccccc',
  :string_quoted_single => ['#ff0000', '#00ff00', 'bold', 'italic']
})
```

Themes in Java

In our Java code we contribute themes via properties files. The key used is the token name and the values are a list of colors and font styles with order being significant. The first color in hex is the fg, the second (optional) is background. We also then take optional font style names such as bold, italic or underline. A few examples:

```
comment.line=#ffff00,#333333,italic
string=#00ff00
invalid.illegal=#521c93,#fffb00,bold
```

Related topics

- Creating a New Theme
- Current Theme Scopes
- User Themes