Lecture 12. Implementing Atom Packages

Cross-Platform Application Development

December 27, 2016

Concepts

Definitions

Atom: a text editor (2014). Features:

- Development by GitHub;
- Open-Source, cross-plaform;
- Based on Electron Framework;
- Uses Node.js runtime and Chromium browser;
- Written in CoffeeScript and Less.

CoffeeScript: a web programming language (2009). Features:

- Source-to-source translated into JavaScript;
- Implementations: Node.js, Java, ...
- Adds syntactic sugar: eliminates the need for parentheses, braces (uses indentation), functional programming style, pattern matching, ...

Using CoffeeScript

Example (test.coffee)

```
polynome = (a, b, c, x) ->
  a * x ** 2 + b * x + c

console.log polynome 1, -1, 1, 2
```

Example

```
> npm install -g coffee-script
...
> echo console.log('AB') | node
AB

> coffee -p test.coffee | node
3
```

Class Example

Example (polynome.coffee)

```
class Polynome
  constructor: (@a, @b, @c) ->

  roots: ->
    D = @b ** 2 -  4 * @a * @c
    return if D == 0 then 1 else if D > 0 then 2 else 0

p = new Polynome 1, -3, 1

console.log p.roots()
```

Using Package Generator

```
${HOME} ......................... or %USERPROFILE%
└─ .atom/
   └─ packages/
      └─ my-package/ ............. package name
         ├─ keymaps/ ........... hot key bindings
         ├─ lib/ ....................... source code
         ├─ menus/ .......... main/context menus
         ├─ spec/ ............................ tests
         ├─ ...
         └─ package.json
```

Figure 1: a directory structure for a simple Atom package

Package Description File

Example (package.json)

```
{
  "name": "my-package",
  "main": "./lib/my-package",
  "version": "0.0.0",
  "description": "A short description of your package",
  "keywords": [
  ],
  "activationCommands": {
    "atom-workspace": "my-package:toggle"
  },
  "repository": "https://github.com/atom/my-package",
  "license": "MIT",
  // ...
```

Example Applications

Method	Comment
`activate(state)`	Called with the serialized state, the workspace is ready
`initialize(state)`	Called before `activate()` and deserialization
`serialize()`	Should return JSON
`deactivate()`	When the window is shutting down

Table 1: entry methods of the package

View Implementation File

Example (my-package-view.js)

```
export default class MyPackageView {

  constructor(serializedState) {
    // Create root element
    this.element = document.createElement('div');
    this.element.classList.add('my-package');

    // Create message element
    const message = document.createElement('div');
    message.textContent = 'The MyPackage package is Alive! It\'s ALIVE!';
    message.classList.add('message');
    this.element.appendChild(message);
  }
```

Logic Implementation File

Example (my-package.js)

```
import MyPackageView from './my-package-view';
import { CompositeDisposable } from 'atom';

export default {

  myPackageView: null,
  modalPanel: null,
  subscriptions: null,
```

Logic Implementation File (cont.)

Example (`my-package.js`, cont.)

```
activate(state) {
  this.myPackageView = new MyPackageView(state.myPackageViewState);
  this.modalPanel = atom.workspace.addModalPanel({
    item: this.myPackageView.getElement(),
    visible: false
  });
```

Logic Implementation File (cont.)

Example (`my-package.js`, cont.)

```
    // Events subscribed to in atom's system can be easily cleaned up
    // with a CompositeDisposable
    this.subscriptions = new CompositeDisposable();

    // Register command that toggles this view
    this.subscriptions.add(atom.commands.add('atom-workspace', {
      'my-package:toggle': () => this.toggle()
    }));
  },

  deactivate() {
    // ...
  }
```

Logic Implementation File (end)

Example (my-package.js, end)

```
// ...

toggle() {
  console.log('MyPackage was toggled!');
  return (
    this.modalPanel.isVisible() ?
    this.modalPanel.hide() :
    this.modalPanel.show()
  );
}
```

Word Count Example

Example (`my-package.coffee`)

```
// ...

toggle: ->
  if @modalPanel.isVisible()
    @modalPanel.hide()
  else
    editor = atom.workspace.getActiveTextEditor()
    words = editor.getText().split(/\s+/).length
    @yourNameWordCountView.setCount(words)
    @modalPanel.show()
```

Word Count Example (end)

Example (my-package-view.coffee)

```
// ...

setCount: (count) ->
  displayText = "There are #{count} words."
  @element.children[0].textContent = displayText
```

Text Modification Example

Example (my-package.coffee)

```
// ...

convert: ->
  if editor = atom.workspace.getActiveTextEditor()
    editor.insertText('Hello, World!')
```