Lecture 11. Introduction to Electron Framework

Cross-Platform Application Development

December 27, 2016

Concepts

Definitions

Electron: a framework for developing desktop applications (2013). Features:

- Development by GitHub;
- Open-Source, cross-plaform;
- Uses Node.js runtime and Chromium browser.

Node.js: a runtime environment for server and web applications (2009). Features:

- Development by Node.js Foundation;
- Open-Source, cross-plaform;
- Asynchronous I/0;
- Event-driven concept;
- Uses Google V8 engine to interpret JavaScript.

Concepts (end)

Definitions

npm: a package manager for Node.js. Features:

- Works with remote repositories (default: `https://registry.npmjs.org/`);
- Installs packages locally (current directory)/globally (system-wide directories);
- Manages the local project dependencies (read from `package.json`);
- Optionally searches for packages with particular version ranges.

Example Applications

Year	Title	Developer
2014	Atom	GitHub
2015	Visual Studio Code	Microsoft
2015	GitKraken	Axosoft

Table 1: applications using Electron framework

Downloading Electron Package

Example

```
> npm install --global electron
...
> electron
```

Hello Example

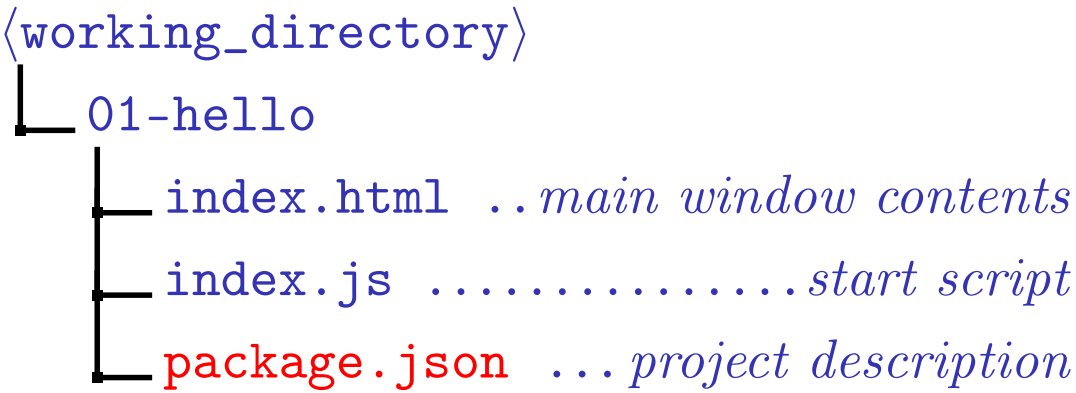

Figure 1: a directory structure for a simple Electron project

Example (package.json)

```
{
  "name": "01-hello",
  "version": "0.0.1",
  "main": "index.js"
}
```

Example (running)

```
> cd 01-hello
> electron .
```

Hello Example (cont.)

Example (index.js)

```
const {app, BrowserWindow} = require('electron')
const path = require('path')
const url = require('url')

let win

function createWindow()
{
  win = new BrowserWindow(
    {
      width: 400,
      height: 200
    })
```

Hello Example (cont.)

Example (index.js, cont.)

```
win.loadURL(
  url.format(
    {
      pathname: path.join(__dirname, 'index.html'),
      protocol: 'file:',
      slashes: true
    }))
// win.webContents.openDevTools()
```

Hello Example (cont.)

Example (index.js, cont.)

```
  win.on(
    'closed',
    () =>
    {
      win = null
    })
}    // createWindow()
```

Example (index.js, cont.)

```
app.on(
  'ready', createWindow)

app.on(
  'window-all-closed',
  () =>
  {
    if (process.platform !== 'darwin')
      app.quit()
  })
```

Hello Example (end)

Example (index.js, end)

```
app.on(
  'activate',
  () =>
  {
    if (win === null)
      createWindow()
  })

// ...
```

Example (index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "UTF-8">
    <title>Hello App</title>
  </head>
  <body>
    <center>
      Hello World!
    </center>
  </body>
</html>
```

Electron Processes

Definitions

Main Process: runs the main script (`index.js`).

Renderer Process: runs each web page with Chromium (`index.html`).

Main Process

- Started/terminated with the whole application;
- Has access to the native GUI API;

Renderer Process

- Started when the main process creates a `BrowserWindow` instance; terminated when it is destroyed;
- Has no access to the native API, isolated with its page.

Debugging the Main Process

Using Electron Inspector

1. `npm install --global node-gyp`
2. On Windows, either:
   - `npm install --global --production windows-build-tools` (admin)
   - or manually:
     1. Install Visual Studio 2015 or Visual C++ Build Tools;
     2. Install Python 2.7
     3. `npm config set python python2.7`
     4. `npm config set msvs_version 2015`
3. `npm install electron-rebuild --save-dev`
4. `npm install electron-inspector --save-dev`
5. `npm install yargs --save-dev`
6. `npm install node-pre-gyp --save-dev`
7. `npm install electron --save-dev`

Debugging the Main Process (end)

Using Electron Inspector

1. Either:
   - `electron --debug=5858 .` or:
   - `electron --debug-brk=5858 .`
2. `node_modules\.bin\electron-inspector`
3. `...\chrome http://127.0.0.1:8080/?port=5858`

Deploying an Application

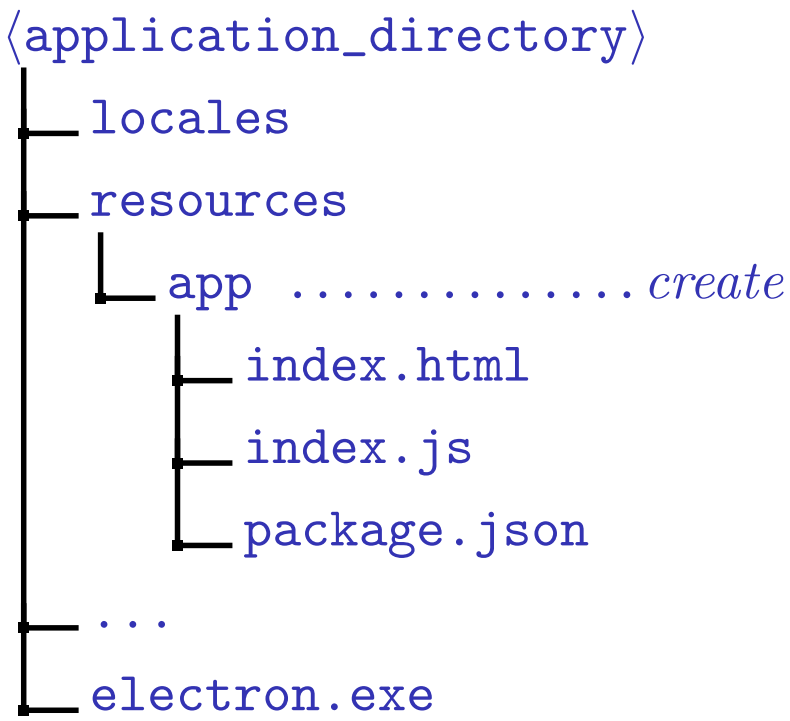

Figure 2: a directory structure for a deployed Electron project

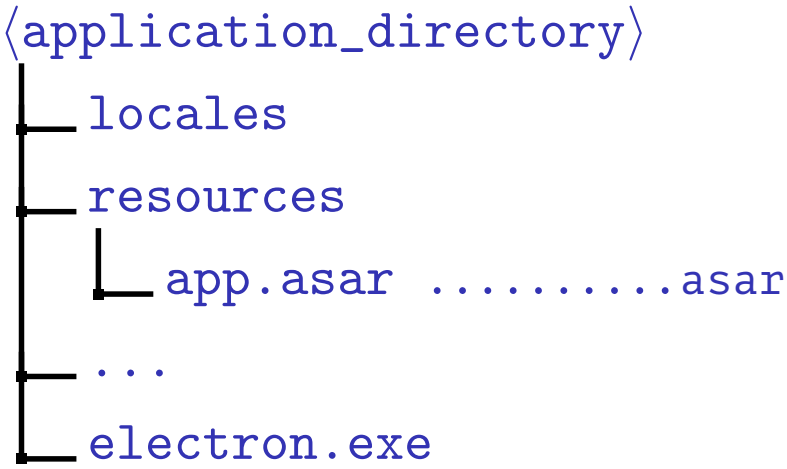

Figure 3: another option for a directory structure

External Module Example

Example (index.js)

```
const {app, BrowserWindow} = require('electron')
const path = require('path')
const url = require('url')

require('./functionality.js')

// ...
```

External Module Example (cont.)

Example (functionality.js)

```
var chalk = require('chalk');

var my_style = chalk.green.bold.underline

chalk.enabled = true

if (chalk.supportsColor)
  console.log(
    chalk.bgRed.dim('Warning: ') + my_style('starting') +
    ' an application')
else
  console.log('No color support')
```

External Module Example (cont.)

Example (console output)

```
> electron .

App threw an error during load
Error: Cannot find module 'chalk'
    at Module._resolveFilename (module.js:455:15)

> npm install --save chalk
```

External Module Example (end)

Example (package.json)

```
{
  "name": "02-module",
  "version": "0.0.1",
  "main": "index.js",
  "dependencies": {
    "chalk": "^1.1.3"
  }
}
```

Example

```
> npm install
```

Managing Windows Example

Example (index.js)

```
function createWindow()
{
  win = new BrowserWindow(
    {
      width: 400,
      height: 200,
      show: false
    })
  win.once(
    'ready-to-show',
    () =>
```

Example (index.js, end)

```
    {
      win.show()
    })
  // ...
  child = new BrowserWindow(
    {
      parent: win,
      modal: true,
      width: 300,
      height: 300
    })
}    // createWindow()
```

Menu Example

Example (index.js)

```
const {app, BrowserWindow, Menu, dialog} = require('electron')
```

Menu Example (cont.)

Example (index.js, cont.)

```
const template =
[
  {
    label: 'File',
    submenu:
    [
      {
        role: 'quit'
      }
    ]
  },
```

Example (index.js, cont.)

```
  {
    label: 'View',
    submenu:
    [
      {
        role: 'reload'
      },
      {
        role: 'toggledevtools'
      }
    ]
  },
```

Menu Example (cont.)

Example (index.js, cont.)

```
{
  label: 'Help',
  submenu:
  [
    {
      label: 'About',
      click()
      {
```

Example (index.js, cont.)

```
          dialog.showMessageBox(
            {
              type: 'info',
              title: 'About my app',
              message: 'My elaborate app',
              buttons: ['OK']
            })
        }    // click()
      }
    ]
  }
]
```

Menu Example (end)

Example (index.js, end)

```
const menu = Menu.buildFromTemplate(template)
Menu.setApplicationMenu(menu)

let win

function createWindow()
{
  // ...
}

// ...
```

Editor Example

Example (index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "UTF-8">
    <title>My Editor</title>
    <style type = "text/css">
      html, body
      {
        height: 100%;
      }
```

Example (index.html, cont)

```
      textarea
      {
        width: 100%;
        height: 90%;
        min-height: 90%;
      }
    </style>
  </head>
```

Editor Example (cont.)

Example (index.html, end)

```
  <body>
    <script src = "./renderer.js"></script>
    <textarea
      id = "editor",
      autofocus = "true",
      oninput = "onInput(this.value)"></textarea>
  </body>
</html>
```

Editor Example (cont.)

Example (renderer.js)

```
const {ipcRenderer} = require('electron')

ipcRenderer.on(
  'opened',
  (event, message) =>
  {
    var editor = document.getElementById('editor')
    editor.innerHTML = message
  })
```

Editor Example (cont.)

Example (renderer.js, end)

```
function onInput(value)
{
  ipcRenderer.send('message', value)
}
```

Editor Example (cont.)

Example (index.js)

```
const {app, BrowserWindow, globalShortcut, ipcMain, dialog} =
  require('electron')
const path = require('path')
const url = require('url')
var fs = require('fs');

var text = '';
```

Editor Example (cont.)

Example (index.js, cont.)

```
ipcMain.on(
  'message',
  (event, arg) =>
  {
    text = arg
  })

let win
```

Example (index.js, cont.)

```
function createWindow()
{
  win = new BrowserWindow(
    {
      width: 800,
      height: 600
    })
```

Editor Example (cont.)

Example (index.js, cont.)

```
win.loadURL(
  url.format(
    {
      pathname: path.join(__dirname, 'index.html'),
      protocol: 'file:',
      slashes: true
    }))
win.on(
  'closed',
  () =>
  {
    win = null
  })
```

Editor Example (cont.)

Example (index.js, cont.)

```
const filters =
[
  {
    name: 'Text Files',
    extensions: ['txt'],
  },
  {
    name: 'All Files',
    extensions: ['*'],
  },
]
```

Example (index.js, cont.)

```
globalShortcut.register(
  'CommandOrControl+S',
  () =>
  {
    dialog.showSaveDialog(
      {
        filters: filters
      },
```

Editor Example (cont.)

Example (index.js, cont.)

```
    (filename) =>
    {
      if (filename != undefined)
        fs.open(
          filename, 'w',
          (err, fd) =>
          {
            if (!err)
              fs.writeSync(fd, text, 0, 'utf-8');
          })
    })    // dialog.showSaveDialog()
})    // globalShortcut.register()
```

Editor Example (cont.)

Example (index.js, cont.)

```
globalShortcut.register(
  'CommandOrControl+O',
  () =>
  {
    dialog.showOpenDialog(
      {
        filters: filters,
        properties: [dialog.openFile]
      },
```

Editor Example (cont.)

Example (index.js, cont.)

```
(filenames) =>
{
  if (filenames != undefined && filenames.length > 0)
    fs.open(
      filenames[0], 'r',
      (err, fd) =>
      {
        var bytesRead = 0
        text = ''
```

Editor Example (cont.)

Example (index.js, cont.)

```
                do
                {
                  const size = 7;
                  var buffer = Buffer.alloc(size)
                  bytesRead = fs.readSync(fd, buffer, 0, size, null)
                  text += buffer
                }
                while (bytesRead > 0)
                win.webContents.send('opened', text)
              })
        })    // dialog.showSaveDialog()
    })    // globalShortcut.register()
}    // createWindow()
```

Tray Example

Example (index.js)

```
const {app, Menu, Tray} = require('electron')

let tray = null

app.on(
  'ready',
  () =>
  {
    tray = new Tray('icon.png')
```

Tray Example (end)

Example (index.js, end)

```
  const contextMenu = Menu.buildFromTemplate(
    [
      {label: 'Item1', type: 'radio', checked: true},
      {label: 'Item2', type: 'radio'},
      {type: 'separator'},
      {label: 'Exit', click() {app.quit()}}
    ])
  tray.setToolTip('My application')
  tray.setContextMenu(contextMenu)
})
```

Network Request Example

Example (index.js)

```
const {app, net} = require('electron')

app.on(
  'ready',
  () =>
  {
    var body = ''
    const request = net.request('https://github.com')
```

Network Request Example (cont.)

Example (index.js, cont.)

```
request.on(
  'response',
  (response) =>
  {
    response.on(
      'data',
      (chunk) =>
      {
        body += chunk
      })
```

Example (index.js, cont.)

```
    response.on(
      'end',
      () =>
      {
        console.log(body)
      })    // response.on()
  })    // request.on()
```

Network Request Example (end)

Example (index.js, cont.)

```
request.on(
  'error',
  (error) =>
  {
    console.log(error)
  })    // request.on()
```

Example (index.js, end)

```
  request.on(
    'close',
    () =>
    {
      console.log('closing')
      app.quit()
    })    // request.on()
  request.end()
})    // app.on()
```