0:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-electron\index.html

```
<html>
  <head>
    <title>Hello World</title>
    <style>
      body {
        background-image: linear-gradient(45deg, #EAD790 0%, #EF8C53 100%);
        text-align: center;
      }

      button {
        background: rgba(0,0,0,0.40);
        box-shadow: 0px 0px 4px 0px rgba(0,0,0,0.50);
        border-radius: 8px;
        color: white;
        padding: 1em 2em;
        border: none;
        font-family: 'Roboto', sans-serif;
        font-weight: 300;
        font-size: 14pt;
        position: relative;
        top: 40%;
        cursor: pointer;
        outline: none;
      }

      button:hover {
        background: rgba(0,0,0,0.30);
      }
    </style>
    <link                href='https://fonts.googleapis.com/css?family=Roboto:300'
rel='stylesheet' type='text/css' />
    <script>
      function sayHello () {
        alert('Hello World');
      }
    </script>
  </head>
  <body>
    <button onclick="sayHello()">Say Hello</button>
  </body>
</html>
```

1:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

2:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-electron\package-lock.json

```
{
    "name": "hello-world",
    "version": "1.0.0",
    "lockfileVersion": 1
}
```

3:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-electron\package.json

```
{
    "name": "hello-world",
    "version": "1.0.0",
    "main": "main.js"
}
```

4:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-nwjs\index.html

```
<html>
    <head>
        <title>Hello World</title>
        <style>
          body {
                background-image: linear-gradient(45deg, #EAD790 0%, #EF8C53 100%);
                text-align: center;
             }

             button {
                background: rgba(0,0,0,0.40);
                box-shadow: 0xp 0px 4px 0px rgba(0,0,0,0.50);
                border-radius: 8px;
                color: white;
                padding: 1em 2em;
                border: none;
                font-family: 'Roboto', sans-serif;
                font-weight: 100;
                font-size: 14pt;
                position: relative;
                top: 40%;
                cursor: pointer;
                outline: none;
             }

             button:hover {
                background: rgba(0,0,0,0.30);
             }
        </style>
        <link           href='https://fonts.googleapis.com/css?family=Roboto:300'
rel='stylesheet' type='text/css'>
        <script>
          function sayHello () {
                alert('Hello World');
             }
        </script>
    </head>
    <body>
        <button onclick="sayHello()">Say Hello</button>
    </body>
</html>
```

5:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-01\hello-world-nwjs\package.json

```
{
  "name": "hello-world-nwjs",
  "main": "index.html",
  "version": "1.0.0"
}
```

6:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-electron\app.js

```
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = {
file: path.basename(filePath),
path: filePath, type: ''
  };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    } else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}

function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function main() {
  let folderPath = getUsersHomeFolder();
  getFilesInFolder(folderPath, (err, files) => {
    if (err) {
      return alert('Sorry, we could not load your home folder');
    }
    inspectAndDescribeFiles(folderPath, files, displayFiles);
  });
}

main();
```

7:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-electron\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder">
        <script>
          document.write(getUsersHomeFolder());
        </script>
      </div>
    </div>
       <div id="main-area"></div>
  </body>
</html>
```

8:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-electron\main.js

```
'use strict';
```

```
const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${app.getAppPath()}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

9:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-electron\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "async": "^2.1.4",
    "osenv": "^0.1.4"
  }
}
```

10:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-nwjs\app.js

```
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = {
file: path.basename(filePath),
path: filePath, type: ''
  };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    } else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}

function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function main() {
  let folderPath = getUsersHomeFolder();
  getFilesInFolder(folderPath, (err, files) => {
    if (err) {
      return alert('Sorry, we could not load your home folder');
    }
    inspectAndDescribeFiles(folderPath, files, displayFiles);
  });
}

main();
```

11:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder">
        <script>
          document.write(getUsersHomeFolder());
        </script>
      </div>
    </div>
       <div id="main-area"></div>
  </body>
</html>
```

12:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-02\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "osenv": "^0.1.4"
  }
}
```

13:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-electron\app.js

```
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

14:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lori
keet-electron\fileSystem.js
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}
```

```
function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

15:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-electron\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

16:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${app.getAppPath()}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

17:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-electron\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  }
}
```

18:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-electron\search.js

```
'use strict';
```

```
const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

19:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lori
keet-electron\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                              class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}
```

```
function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };
```

20:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-nwjs\app.js

```
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;
```

21:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-nwjs\fileSystem.js

```
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');
```

```
let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

22:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
                <div id="current-folder"></div>
                <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

23:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  }
}
```

24:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lorikeet-nwjs\search.js

```
'use strict';

const lunr = require('lunr');
let index;
```

```
function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

25:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-03\lori
keet-nwjs\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}
```

```
function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}
```

```
function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };

26:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-electron\app.js
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

27:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-electron\fileSystem.js
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;
```

```
if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

28:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-electron\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

29:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${app.getAppPath()}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

30:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-electron\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "main.js",
     "author": "Paul Jensen <paul@anephenix.com>",
     "description": "A file explorer application",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  },
     "scripts" : {
          "pack": "build",
          "dist": "build"
     },
  "devDependencies": {
    "electron": "^1.4.14",
    "electron-builder": "^11.4.4"
  },
     "build": {}
}
```

31:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-electron\search.js

```
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

32:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-electron\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
        search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports  =  {  bindDocument,  displayFiles,  loadDirectory,  bindSearchField,
filterResults, resetFilter };

33:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-nwjs\app.js
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');
```

```
function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

34:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-nwjs\fileSystem.js
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

35:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
              <div id="current-folder"></div>
```

```
          <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

36:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  }
}
```

37:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lorikeet-nwjs\search.js

```
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}
```

```
module.exports = { addToIndex, find, resetIndex };

38:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-04\lori
keet-nwjs\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
        search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };

39:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\cons
trained-window-sizing-electron\index.html
<html>
  <head>
    <title>Constrained window sizing Electron</title>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

40:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\constrained-window-sizing-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 400, height: 200,
    minWidth: 300, minHeight: 150,
    maxWidth: 600, maxHeight: 450
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

41:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\constrained-window-sizing-electron\package.json

```
{
  "name"    : "constrained-window-sizing-electron",
  "version" : "1.0.0",
  "main"    : "main.js"
}
```

42:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-electron\index.html

```
<html>
  <head>
    <title>Dynamic window positioning Electron</title>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

43:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 400, height: 200,
    x: 10, y: 10
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

44:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-electron\package.json

```
{
  "name"    : "dynamic-window-positioning-electron",
  "version" : "1.0.0",
  "main"    : "main.js"
}
```

45:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-nwjs\app.js

```
const gui = require('nw.gui');
const win = gui.Window.get();
win.x = 400;
win.y = 500;
```

46:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-nwjs\index.html

```
<html>
  <head>
    <title>Dynamic window positioning NW.js</title>
    <script src="app.js"></script>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

47:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-positioning-nwjs\package.json

```
{
  "name" : "dynamic-window-positioning-nwjs",
  "version" : "1.0.0",
  "main" : "index.html",
  "window" : {
     "width" : 300,
     "height" : 200
  }
}
```

48:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-sizing-nwjs\app.js

```
const gui = require('nw.gui');
const win = gui.Window.get();

win.width = 1024;
win.height = 768;
```

49:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-sizing-nwjs\index.html

```
<html>
  <head>
    <title>Dynamic window sizing NW.js</title>
    <script src="app.js"></script>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

50:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\dynamic-window-sizing-nwjs\package.json

```
{
  "name" : "dynamic-window-sizing-nwjs",
  "version" : "1.0.0",
  "main" : "index.html"
}
```

51:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\frameless-app-nwjs\index.html

```
<html>
  <head>
```

```
    <title>Transparent NW.js app - you won't see this title</title>
    <style rel="stylesheet">
      html {
        border-radius: 25px;
        -webkit-app-region: drag;
       }

      body {
        background: #333;
        color: white;
        font-family: 'Signika';
      }

      p {
        padding: 1em;
        text-align: center;
        text-shadow: 1px 1px 1px rgba(0,0,0,0.25);
      }

       button, select {
         -webkit-app-region: no-drag;
       }

       p, img {
         -webkit-user-select: all;
         -webkit-app-region: no-drag;
       }

    </style>
  </head>
  <body>
    <p>Frameless app example</p>
  </body>
</html>
```

52:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\frameless-app-nwjs\package.json

```
{
     "name" : "frameless-transparent-app-nwjs",
     "version" : "1.0.0",
     "main" : "index.html",
     "window" : {
       "frame" : false,
       "transparent": true,
       "width": 300,
       "height": 150
     }
}

53:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\full
screen-app-electron\app.js
const remote = require('electron').remote;

function toggleFullScreen() {
     const button = document.getElementById('fullscreen');
     const win = remote.getCurrentWindow();
  if (win.isFullScreen()) {
    win.setFullScreen(false);
    button.innerText = 'Go full screen';
  } else {
    win.setFullScreen(true);
    button.innerText = 'Exit full screen';
  }
}

54:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\full
screen-app-electron\index.html
<html>
  <head>
    <title>Fullscreen app programmatic Electron</title>
  </head>
     <script src="app.js"></script>
  <body>
     <h1>Hello from Electron</h1>
     <button id="fullscreen" onclick="toggleFullScreen();">
          Go full screen
          </button>
  </body>
</html>

55:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\full
screen-app-electron\main.js
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

```
app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

56:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\fullscreen-app-electron\package.json

```
{
  "name"    : "fullscreen-app-electron",
  "version" : "1.0.0",
  "main"    : "main.js"
}
```

57:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\fullscreen-app-nwjs\index.html

```
<html>
  <head>
    <title>Full-screen app example</title>
    <script>
      'use strict';
      const gui = require('nw.gui');
      const win = gui.Window.get();

      function toggleFullScreen () {
        const button = document.getElementById('fullscreen');
        if (win.isFullscreen) {
          win.leaveFullscreen();
          button.innerText = 'Go full screen';
        } else {
          win.enterFullscreen();
          button.innerText = 'Exit full screen';
        }
      }

    </script>
  </head>
  <body>
    <h1>Full-screen app example</h1>
    <button id="fullscreen" onclick="toggleFullScreen();">Go full screen</button>
  </body>
</html>
```

58:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\fullscreen-app-nwjs\package.json

```
{
     "name":"fullscreen-app-nwjs",
     "version":"1.0.0",
     "main":"index.html"
}
```

59:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\window-sizing-electron\index.html

```
<html>
  <head>
    <title>Window sizing Electron</title>
  </head>
  <body>
    <h1>Hello from Electron</h1>
  </body>
</html>
```

60:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\window-sizing-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({ width: 400, height: 200 });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

61:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\window-sizing-electron\package.json

```
{
  "name"    : "window-sizing-electron",
  "version" : "1.0.0",
  "main"    : "main.js"
}
```

62:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\window-sizing-nwjs\index.html

```
<html>
  <head>
```

```
    <title>Window sizing NW.js</title>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

63:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-07\window-sizing-nwjs\package.json

```
{
  "name" : "window-sizing-nwjs",
  "version" : "1.0.0",
  "main" : "index.html",
  "window" : {
     "width" : 300,
     "height" : 200
  }
}
```

64:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-electron\app.js

```
function displayNote(event, note) {
  document.getElementById('title').innerText = note.title;
  document.getElementById('contents').innerText = note.contents;
}

const ipc = require('electron').ipcRenderer;
ipc.on('displayNote', displayNote);
```

65:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-electron\index.html

```
<html>
  <head>
    <title>Tray app Electron</title>
    <link href="app.css" rel="stylesheet">
    <script src="app.js"></script>
  </head>
  <body>
    <h1 id="title"></h1>
    <div id="contents"></div>
  </body>
</html>
```

66:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const Menu = electron.Menu;
const Tray = electron.Tray;
const BrowserWindow = electron.BrowserWindow;

let appIcon = null;
let mainWindow = null;

const notes = [
  {
    title: 'todo list',
    contents: 'grocery shopping\npick up kids\nsend birthday party invites'
  },
  {
    title: 'grocery list',
    contents: 'Milk\nEggs\nButter\nDouble Cream'
  },
  {
    title: 'birthday invites',
    contents: 'Dave\nSue\nSally\nJohn and Joanna\nChris and Georgina\nElliot'
  }
];

function displayNote (note) {
  mainWindow.webContents.send('displayNote', note);
}

function addNoteToMenu (note) {
  return {
    label: note.title,
    type: 'normal',
    click: () => { displayNote(note); }
  };
}

app.on('ready', () => {
  appIcon = new Tray('icon@2x.png');
  let contextMenu = Menu.buildFromTemplate(notes.map(addNoteToMenu));
  appIcon.setToolTip('Notes app');
  appIcon.setContextMenu(contextMenu);

  mainWindow = new BrowserWindow({ width: 800, height: 600 });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.webContents.on('dom-ready', () => {
    displayNote(notes[0]);
  });
});
```

67:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-electron\package.json

```
{
  "name"    : "tray-app-electron",
  "version" : "1.0.0",
  "main"    : "main.js"
}
```

68:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-nwjs\index.html

```
<html>
  <head>
    <title>Tray app example</title>
          <link href="app.css" rel="stylesheet">
    <script>
              'use strict';
              const notes = [
                {
                  title: 'todo list',
                   contents: 'grocery shopping\npick up kids\nsend birthday party
invites'},
                {
                  title: 'grocery list',
                  contents: 'Milk\nEggs\nButter\nDouble Cream'},
                {
                  title: 'birthday invites',
                       contents:  'Dave\nSue\nSally\nJohn  and  Joanna\nChris  and
Georgina\nElliot'
                }
              ];

              function displayNote (note) {
             document.getElementById('title').innerText = note.title;
             document.getElementById('contents').innerText = note.contents;
           }

      const gui = require('nw.gui');
      const tray = new gui.Tray({icon: 'icon@2x.png'});
              const menu = new gui.Menu();

              function appendNoteToMenu (note) {
                const menuItem = new gui.MenuItem({
                  label: note.title,
                  click: () => { displayNote(note); }
                });
                menu.append(menuItem);
              }

              notes.map(appendNoteToMenu);

              document.addEventListener('DOMContentLoaded', () => {
                displayNote(notes[0]);
              });

              tray.menu = menu;
    </script>
  </head>
  <body>
        <h1 id="title"></h1>
    <div id="contents"></div>
  </body>
</html>
```

69:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-08\tray-app-nwjs\package.json

```
{
     "name": "tray-app-nwjs",
     "version": "1.0.0",
  "main":"index.html"
}
```

70:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-electron\app.js

```
'use strict';

// Dependencies
//
const electron          = require('electron');
const Menu         = electron.remote.Menu;
const MenuItem     = electron.remote.MenuItem;
const ipc          = electron.ipcRenderer;
const dialog       = electron.remote.dialog;
const designMenu   = require('./designMenu');
let currentFile;
let content;
let tabWas;
let done;

ipc.on('fileRead', (event, err, data) => {
     loadMenu(true);
     if (err) throw(err);
     if (!done) bindClickingOnTabs();
     hideSelectFileButton();
     setContent(data);
     showViewMode('design');
});

ipc.on('fileSaved', (event, err) => {
     if (err) return alert('There was an error saving the file');
     alert('File Saved');
});

function openFile (cb) {
     dialog.showOpenDialog((files) => {
          ipc.send('readFile', files);
          if (files) currentFile = files[0];
          if (cb && typeof cb === 'function') cb();
     });
}

function saveFile () {
     ipc.send('saveFile', currentFile, content);
}

function loadMenu (enableSaveOption) {
     const template = [
       {
         label: 'File',
         submenu: [
           {
             label: 'Open File',
             click: openFile
           }
         ]
       }
     ];

     if (enableSaveOption) {
          template[0].submenu.push({
               label: 'Save File',
               click: saveFile
          });
     }

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

```
}

function bindSelectFileClick (cb) {
     const button = document.querySelector('#openFileView div');
     button.addEventListener('click', () => {
          openFile(cb);
     });
}

function hideSelectFileButton () {
     const button = document.querySelector('#openFileView');
     button.classList.add('hidden');
     const appView = document.querySelector('#appView');
     appView.classList.remove('hidden');
}

function hideDiv (div) {
     if (!div.classList.contains('hidden')) div.classList.add('hidden');
}

function showViewMode (viewMode) {
     const areaDivs = document.querySelectorAll('.area');
     areaDivs.forEach(hideDiv);
     const selectedArea = document.querySelector(`#${viewMode}Area`);
     selectedArea.classList.remove('hidden');
     tabWas = viewMode;
}

function setContent (changedContent) {
     if (changedContent) { content = changedContent; }
     const designArea = document.querySelector('#designArea');
     designArea.innerHTML = content;
     const codeArea = document.querySelector('#codeArea');
     codeArea.value = content;
     const previewArea = document.querySelector('#previewArea');
     previewArea.innerHTML = content;
}

function bindClickingOnTab (tabDiv) {
     tabDiv.addEventListener('click', () => {
          const id = tabDiv.id;
          if (tabWas) {
               const contentDiv = document.querySelector(`#${tabWas}Area`);
               if (tabWas === 'design') setContent(contentDiv.innerHTML);
               if (tabWas === 'code') setContent(contentDiv.value);
          }
          showViewMode(id);
```

```
    });
}

function bindClickingOnTabs() {
    const tabs = document.querySelectorAll('.tab');
    done = true;
    tabs.forEach(bindClickingOnTab);
}

function bindOnDesignView() {
    designMenu();
}

function initialize () {
    loadMenu();
    bindSelectFileClick();
    bindOnDesignView();
}

window.onload = initialize;

71:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirr
us-electron\designMenu.js
'use strict';


var electron = require('electron');
var Menu          = electron.remote.Menu;
var MenuItem  = electron.remote.MenuItem;
var Dialogs       = require('dialogs');
var dialogs = new Dialogs();

// Used to store the coordinates where
// the context menu was clicked
var x,y;



function insertContent (content) {
    var range = document.caretRangeFromPoint(x, y);
    if (range) {
      range.insertNode(content);
    }
}


function openImageFileDialog (cb) {
    var inputField = document.querySelector('#imageFileSelector');
    inputField.addEventListener('change', function () {
        var filePath = this.value;
        cb(filePath);
    });
    inputField.click();
}



function insertImage () {
    openImageFileDialog(function (filePath) {
        if (filePath !== '') {
            var newImageNode = document.createElement('img');
            newImageNode.src = filePath;
            insertContent(newImageNode);
        }
    });
}



function parseYoutubeVideo (youtubeURL) {
    if (youtubeURL.indexOf('youtube.com/watch?v=') > -1) {
        return youtubeURL.split('watch?v=')[1];
    } else if (youtubeURL.match('https://youtu.be/') !== null) {
        return youtubeURL.split('https://youtu.be/')[1];
    } else if (youtubeURL.match('<iframe') !== null) {
        return youtubeURL.split('youtube.com/embed/')[1].split('"')[0];
    } else {
        alert('Unable to find a YouTube video id in the url');
        return false;
    }
}



function insertVideo () {
    dialogs.prompt('Please insert a YouTube url', (youtubeURL) => {
        if (youtubeURL) {
            var videoId = parseYoutubeVideo(youtubeURL);
            if (videoId) {
                var newIframeNode = document.createElement('iframe');
                newIframeNode.width = 854;
                newIframeNode.height = 480;
                newIframeNode.src = 'https://www.youtube.com/embed/' + videoId;
                newIframeNode.frameborder = 0;
                newIframeNode.allowfullscreen = true;
                setTimeout(() => {
                    insertContent(newIframeNode);
                }, 300);
            }
        }
    });
}



function initialize () {
    const menu = new Menu();

    menu.append(new MenuItem({label: 'Insert image', click: insertImage }));
    menu.append(new MenuItem({label: 'Insert video', click: insertVideo }));

    document.querySelector('#designArea')
    .addEventListener('contextmenu', function (event) {
        event.preventDefault();
        x = event.x;
        y = event.y;
        menu.popup(event.x, event.y);
        return false;
    });

}



module.exports = initialize;
```

72:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-electron\index.html

```
<html>
  <head>
    <title>Cirrus</title>
    <link href="app.css" rel="stylesheet" />
    <script src="app.js"></script>
  </head>
  <body>
        <input      type="file"      accept="image/*"      id="imageFileSelector"
class="hidden"/>
    <input type="file" accept=".html,.htm" id="fileSelector" class="hidden"/>
        <div id="openFileView">
            <div>Select a HTML file</div>
        </div>
		<div id="appView" class="hidden">
			<div id="toolbar">
				<div class="tab" id="design">Design</div>
				<div class="tab" id="code">Code</div>
				<div class="tab" id="preview">Preview</div>
			</div>
			<div class="area hidden" id="designArea" contenteditable></div>
			<textarea class="area hidden" id="codeArea"></textarea>
			<div class="area hidden" id="previewArea"></div>
		</div>
  </body>
</html>
```

73:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-electron\main.js

```
'use strict';

const electron = require('electron');
const fs = require('fs');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;
const ipc = electron.ipcMain;
let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});

function readFile (event, files) {
  if (files) {
    // We can only load one file in the app, so we select the first
    const filePath = files[0];
     fs.readFile(filePath, 'utf8', (err, data) => {
       event.sender.send('fileRead', err, data);
     });
  }
};

function saveFile (event, currentFile, content) {
  fs.writeFile(currentFile, content, (err) => {
		event.sender.send('fileSaved', err);
```

```
    });
}

// Handles reading the contents of a file
ipc.on('readFile', readFile);
ipc.on('saveFile', saveFile);
```

74:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-electron\package.json

```
{
  "name": "Cirrus",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "dialogs": "^1.1.17",
    "electron-prebuilt": "^1.2.2"
  },
  "scripts": {
    "start": "node_modules/.bin/electron ."
  }
}
```

75:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-nwjs\app.js

```
'use strict';



// Dependencies
//
const fs           = require('fs');
const gui          = require('nw.gui');
const designMenu   = require('./designMenu');
let currentFile;
let content;


function openFile () {
     openFileDialog((filePath) => {
          fs.readFile(filePath, (err, data) => {
               setContent(data);
               hideSelectFileButton();
               showViewMode('design');
          });
     });
}
```

```
function saveFile () {
     fs.writeFile(currentFile, content, (err) => {
          if (err) {
               alert('There was an error');
          }
     });
}



function loadMenu () {

     const menuBar = new gui.Menu({type:'menubar'});

     // Create sub-menu
     const menuItems = new gui.Menu();

     menuItems.append(new gui.MenuItem({ label: 'Open', click: openFile }));
     menuItems.append(new gui.MenuItem({ label: 'Save', click: saveFile }));


     if (process.platform === 'darwin') {

          // Load Mac OS X application menu
          menuBar.createMacBuiltin('Cirrus');

          menuBar.insert(
              new gui.MenuItem({
                  label: 'File',
                  submenu: menuItems // menu elements from menuItems object
              }), 1
          );

     } else {

          // Load Windows/Linux application menu
          menuBar.append(
              new gui.MenuItem({
                  label: 'File',
                  submenu: menuItems // menu elements from menuItems object
              }), 1
          );

     }
```

```
    gui.Window.get().menu = menuBar;

}



function openFileDialog (cb) {
    const inputField = document.querySelector('#fileSelector');
    inputField.addEventListener('change', function () {
        const filePath = this.value;
        currentFile = filePath;
        cb(filePath);
    });
    inputField.click();
}



function bindSelectFileClick (cb) {
    const button = document.querySelector('#openFileView div');
    button.addEventListener('click', () => {
        openFileDialog(cb);
    });
}



function hideSelectFileButton () {
    const button = document.querySelector('#openFileView');
    button.classList.add('hidden');
    const appView = document.querySelector('#appView');
    appView.classList.remove('hidden');
}



function showViewMode (viewMode) {
    const areaDivs = document.querySelectorAll('.area');
    for (let i=0;i<areaDivs.length;i++) {
        let areaDiv = areaDivs[i];
        areaDiv.classList.add('hidden');
    }
    const selectedArea = document.querySelector(`#${viewMode}Area`);
    selectedArea.classList.remove('hidden');
}
```

```
function setContent (changedContent) {
     if (changedContent) { content = changedContent; }
     const designArea = document.querySelector('#designArea');
     designArea.innerHTML = content;
     const codeArea = document.querySelector('#codeArea');
     codeArea.value = content;
     const previewArea = document.querySelector('#previewArea');
     previewArea.innerHTML = content;
}

function initialize () {
     bindSelectFileClick((filePath) => {
          loadMenu();
          fs.readFile(filePath, (err, data) => {
               setContent(data);
               hideSelectFileButton();
               showViewMode('design');
          });
     });
     designMenu(window, gui);
}



window.onload = initialize;
```

76:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-nwjs\designMenu.js

```
'use strict';

let x;
let y;
let document;

function insertContent (content) {
     const range = document.caretRangeFromPoint(x, y);
     if (range) {
       range.insertNode(content);
     }
}

function openImageFileDialog (cb) {
     const inputField = document.querySelector('#imageFileSelector');
     inputField.addEventListener('change', () => {
          const filePath = this.value;
          cb(filePath);
```

```
    });
    inputField.click();
}

function insertImage () {
    openImageFileDialog((filePath) => {
        if (filePath !== '') {
            const newImageNode = document.createElement('img');
            newImageNode.src = filePath;
            insertContent(newImageNode);
        }
    });
}

function parseYoutubeVideo (youtubeURL) {
    if (youtubeURL.indexOf('youtube.com/watch?v=') > -1) {
        return youtubeURL.split('watch?v=')[1];
    } else if (youtubeURL.match('https://youtu.be/') !== null) {
        return youtubeURL.split('https://youtu.be/')[1];
    } else if (youtubeURL.match('<iframe') !== null) {
        return youtubeURL.split('youtube.com/embed/')[1].split('"')[0];
    } else {
        alert('Unable to find a YouTube video id in the url');
        return false;
    }
}

function insertVideo () {
    const youtubeURL = prompt('Please insert a YouTube url');
    if (youtubeURL) {
        const videoId = parseYoutubeVideo(youtubeURL);
        if (videoId) {
            const newIframeNode = document.createElement('iframe');
            newIframeNode.width = 854;
            newIframeNode.height = 480;
            newIframeNode.src = 'https://www.youtube.com/embed/' + videoId;
            newIframeNode.frameborder = 0;
            newIframeNode.allowfullscreen = true;
            insertContent(newIframeNode);
        }
    }
}

function initialize (window, gui) {

    if (!document) document = window.document;
    const menu = new gui.Menu();

    menu.append(new gui.MenuItem({icon: 'picture.png', label: 'Insert image', click:
insertImage }));
    menu.append(new gui.MenuItem({icon: 'youtube.png', label: 'Insert video', click:
insertVideo }));

    document.querySelector('#designArea')
    .addEventListener('contextmenu', (event) => {
        event.preventDefault();
        x = event.x;
        y = event.y;
        menu.popup(event.x, event.y);
        return false;
    });

}

module.exports = initialize;

77:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirr
us-nwjs\index.html
<!doctype html>
<html lang="en">
    <head>
        <title>Cirrus</title>
        <link href="app.css" rel="stylesheet" />
        <script src="app.js"></script>
    </head>
    <body>
        <input      type="file"      accept="image/*"      id="imageFileSelector"
class="hidden"/>
        <input type="file" accept=".html,.htm" id="fileSelector" class="hidden"/>
        <div id="openFileView">
            <div>Select a HTML file</div>
        </div>
        <div id="appView" class="hidden">
            <div id="toolbar">
                <div                  class="tab"                  id="design"
onclick="showViewMode('design');">Design</div>
                <div                  class="tab"                  id="code"
onclick="showViewMode('code');">Code</div>
                <div                  class="tab"                  id="preview"
onclick="showViewMode('preview');">Preview</div>
            </div>
            <div    class="area    hidden"    id="designArea"    contenteditable
onblur="setContent(this.innerHTML);"></div>
```

```
            <textarea        class="area        hidden"        id="codeArea"
onblur="setContent(this.value);"></textarea>
            <div class="area hidden" id="previewArea"></div>
        </div>
    </body>
</html>
```

78:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-nwjs\package.json

```
{
    "name":"cirrus",
    "version":"1.0.0",
    "main":"index.html",
    "window":{
        "icon":"cirrus-logo.png",
        "toolbar":false
    }
}
```

79:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\cirrus-nwjs\README.md

```
# Cirrus (NW.js)
A WYSIWYG HTML editor, built with NW.js

### Installation

    npm install -g nw
    cd cirrus
    nw

### About Cirrus

This is the source code for one of the apps featured in ["Cross Platform Desktop Applications"](http://manning.com/books/cross-platform-desktop-applications).
```

80:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-default-menu-electron\app.js

```
'use strict';

const electron = require('electron');
const defaultMenu = require('electron-default-menu');
const Menu  = electron.remote.Menu;

const menu = Menu.buildFromTemplate(defaultMenu());
Menu.setApplicationMenu(menu);
```

81:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-

app-default-menu-electron\index.html
```
<html>
     <head>
          <title>Mac App Menu example</title>
          <script src="app.js"></script>
     </head>
     <body>
          <h1>Electron Mac App Menu example</h1>
     </body>
</html>
```

82:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-default-menu-electron\main.js
```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

83:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-default-menu-electron\package.json
```
{
  "name": "mac-app-default-menu-electron",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "electron-default-menu": "^1.0.0"
  }
}
```

84:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-electron\app.js
```
'use strict';

const electron = require('electron');
const Menu  = electron.remote.Menu;
const name = electron.remote.app.getName();

const template = [{
     label: '',
     submenu: [
       {
         label: 'About ' + name,
         role: 'about'
       },
    {
      type: 'separator'
    },
    {
      label: 'Quit',
      accelerator: 'Command+Q',
      click: electron.remote.app.quit
    }
     ]
}];

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

85:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-electron\index.html

```
<html>
     <head>
          <title>Mac App Menu example</title>
          <script src="app.js"></script>
     </head>
     <body>
          <h1>Electron Mac App Menu example</h1>
     </body>
</html>
```

86:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

87:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-electron\package.json

```
{
     "name":"mac-app-menu-electron",
     "version":"1.0.0",
     "main":"main.js"
}
```

88:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-nwjs\app.js

```
'use strict';

const gui = require('nw.gui');

const mb = new gui.Menu({ type: 'menubar' });
mb.createMacBuiltin('Mac app menu example');

gui.Window.get().menu = mb;
```

89:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-nwjs\index.html

```
<html>
     <head>
          <title>Mac app menu NW.js</title>
          <script src="app.js"></script>
     </head>
     <body>
          <h1>Hello world</h1>
     </body>
</html>
```

90:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\mac-app-menu-nwjs\package.json

```
{
     "name":"map-app-menu-nwjs",
     "version":"1.0.0",
     "main":"index.html"
}
```

91:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-electron\app.js

```
'use strict';

const electron = require('electron');
const Menu  = electron.remote.Menu;

const sayHello = () => { alert('Hello'); };

const quitTheApp = () => { electron.remote.app.quit(); };

const template = [
  {
    label: 'File',
    submenu: [
      {
        label: 'Say Hello',
        click: sayHello
      },
      {
        label: 'Quit the app',
        click: quitTheApp
      }
    ]
  }
];

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

92:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-electron\index.html

```
<html>
  <head>
    <title>Windows/Linux menu app example for Electron</title>
          <script src="app.js"></script>
  </head>
  <body>
    <h1>Windows/Linux menu example</h1>
  </body>
</html>
```

93:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-electron\main.js

```
'use strict';
```

```
const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

94:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-electron\package.json

```
{
     "windows-linux-menu-app-electron",
     "version":"1.0.0",
     "main":"main.js"
}
```

95:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-nwjs\index.html

```
<html>
  <head>
    <title>Windows/Linux menu app example for NW.js</title>
          <script>
      'use strict';

      const gui       = require('nw.gui');
      const menuBar   = new gui.Menu({type:'menubar'});
      const fileMenu  = new gui.MenuItem({label: 'File'});

              const sayHelloMenuItem = new gui.MenuItem(
                {
                  label: 'Say hello',
                  click: () => { alert('Hello'); }
                }
              );

              const quitAppMenuItem = new gui.MenuItem(
                {
                  label: 'Quit the app',
                  click: () => { process.exit(0); }
```

```
            }
          );

      const fileMenuSubMenu = new gui.Menu();
      fileMenuSubMenu.append(sayHelloMenuItem);
      fileMenuSubMenu.append(quitAppMenuItem);

      fileMenu.submenu = fileMenuSubMenu;

      menuBar.append(fileMenu);
      gui.Window.get().menu = menuBar;
    </script>
  </head>
  <body>
    <h1>Windows/Linux menu example</h1>
  </body>
</html>
```

96:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-09\windows-linux-menu-app-nwjs\package.json

```
{
     "name":"windows-linux-app-menu-nwjs",
     "version":"1.0.0",
     "main":"index.html"
}
```

97:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-electron\app.js

```
'use strict';


function addStylesheet (stylesheet) {
  var head = document.getElementsByTagName('head')[0];
  var link = document.createElement('link');
  link.setAttribute('rel','stylesheet');
  link.setAttribute('href',stylesheet+'.css');
  head.appendChild(link);
}

function labelOS (osName) {
  document.getElementById('os-label').innerText = osName;
}

function initialize () {
  var os      = require('os');
  var platform     = os.platform();
  switch (platform) {
     case 'darwin':
       addStylesheet('mac');
      labelOS('macOS');
       break;
     case 'linux':
       addStylesheet('linux');
      labelOS('Linux');
       break;
     case 'win32':
       addStylesheet('windows');
      labelOS('Microsoft Windows');
       break;
     default:
       console.log('Could not detect OS for platform',platform);
  }
}

window.onload = initialize;
```

98:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-electron\index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Detect OS (Electron)</title>
<link rel="stylesheet" href="app.css">
<script src="app.js">
</script>
</head>
<body>
<p>You are running <span id="os-label">(OS)</span></p>
</body>
</html>
```

99:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-electron\main.js

```
'use strict';

var electron = require('electron');
var app = electron.app;
var BrowserWindow = electron.BrowserWindow;
var mainWindow = null;

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

```
app.on('ready', function () {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL('file://' + __dirname + '/index.html');
  mainWindow.on('closed', function () { mainWindow = null; });
     //mainWindow.webContents.openDevTools();
});
```

100:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-electron\package.json

```
{
  "name": "detect-os",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "electron-prebuilt": "^1.2.2"
  },
  "scripts": {
    "start": "node_modules/.bin/electron ."
  }
}
```

101:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-nwjs\app.js

```
'use strict';

var os        = require('os');
var platform  = os.platform();


function addStylesheet (stylesheet) {
  var head = document.getElementsByTagName('head')[0];
  var link = document.createElement('link');
  link.setAttribute('rel','stylesheet');
  link.setAttribute('href',stylesheet+'.css');
  head.appendChild(link);
}

function labelOS (osName) {
  document.getElementById('os-label').innerText = osName;
}

function initialize () {
  switch (platform) {
     case 'darwin':
       addStylesheet('mac');
      labelOS('macOS');
```

```
        break;
      case 'linux':
        addStylesheet('linux');
       labelOS('Linux');
        break;
      case 'win32':
        addStylesheet('windows');
       labelOS('Microsoft Windows');
        break;
      default:
        console.log('Could not detect OS for platform',platform);
  }
}

window.onload = initialize;
```

102:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-nwjs\index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Detect OS (NW.js)</title>
<link rel="stylesheet" href="app.css">
<script src="app.js">
</script>
</head>
<body>
<p>You are running <span id="os-label">(OS)</span></p>
</body>
</html>
```

103:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\detect-os-nwjs\package.json

```
{
  "name": "detect-os",
  "version": "1.0.0",
  "main": "index.html",
  "scripts": {
    "start": "node_modules/.bin/nw ."
  },
  "dependencies": {
    "nw": "^0.15.2"
  }
}
```

104:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-electron\app.js

```
'use strict';

function stopDefaultEvent (event) {
     event.preventDefault();
     return false;
}

window.ondragover = stopDefaultEvent;
window.ondrop = stopDefaultEvent;

function displayImageInIconSet (filePath) {
     var images = window.document.querySelectorAll('#icons img');
     for (var i=0;i < images.length;i++) {
          images[i].src = filePath;
     }
}

function displayIconsSet () {
     var iconsArea = window.document.querySelector('#icons');
     iconsArea.style.display = 'block';
}

function interceptDroppedFile () {
     var interceptArea = window.document.querySelector('#load-icon-holder');
     interceptArea.ondrop = function (event) {
          event.preventDefault();
          if (event.dataTransfer.files.length !== 1) {
               window.alert('You have dragged too many files into the app. Drag just
1 file');
          } else {
               interceptArea.style.display = 'none';
               displayIconsSet();
               var file = event.dataTransfer.files[0];
               displayImageInIconSet(file.path);
          }
          return false;
     };
}

window.onload = function () {
     interceptDroppedFile();
};
```

105:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\ico
nic-electron\index.html

```
<html>
     <head>
```

```
        <title>Iconic</title>
        <link rel="stylesheet" href="app.css" />
        <script src="app.js"></script>
    </head>
    <body>
        <div id="load-icon-holder">
            <h1>Drag and Drop your file here</h1>
            <img src="images/drop-here.png" />
        </div>
        <div id="icons">
            <div class="icon-holder">
                <label>16x16</label>
                <img class="icon sixteen" />
            </div>
            <div class="icon-holder">
                <label>32x32</label>
                <img class="icon thirtytwo" />
            </div>
            <div class="icon-holder">
                <label>64x64</label>
                <img class="icon sixtyfour" />
            </div>
            <div class="icon-holder">
                <label>128x128</label>
                <img class="icon onetwoeight" />
            </div>
            <div class="icon-holder">
                <label>256x256</label>
                <img class="icon twofivesix" />
            </div>
            <div id="save">
                <p>Click on an image to save it to your computer</p>
            </div>
        </div>
    </body>
</html>
```

106:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-electron\main.js

```
'use strict';

var electron = require('electron');
var app = electron.app;
var BrowserWindow = electron.BrowserWindow;
var mainWindow = null;

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

app.on('ready', function () {
  mainWindow = new BrowserWindow({width: 650, height: 510});
  mainWindow.loadURL('file://' + __dirname + '/index.html');
  mainWindow.on('closed', function () { mainWindow = null; });
     //mainWindow.webContents.openDevTools();
});
```

107:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-electron\package.json

```
{
  "name": "iconic",
  "version": "1.0.0",
  "main": "main.js",
  "dependencies": {
    "electron-prebuilt": "^1.2.2"
  },
  "scripts": {
    "start": "node_modules/.bin/electron ."
  }
}
```

108:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-nwjs\app.js

```
'use strict';

function stopDefaultEvent (event) {
     event.preventDefault();
     return false;
}

window.ondragover = stopDefaultEvent;
window.ondrop = stopDefaultEvent;

function displayImageInIconSet (filePath) {
     var images = window.document.querySelectorAll('#icons img');
     for (var i=0;i<images.length;i++) {
          images[i].src = filePath;
     }
}

function displayIconsSet () {
     var iconsArea = window.document.querySelector('#icons');
     iconsArea.style.display = 'block';
}
```

```
function interceptDroppedFile () {
     var interceptArea = window.document.querySelector('#load-icon-holder');
     interceptArea.ondrop = function (event) {
          event.preventDefault();
          if (event.dataTransfer.files.length !== 1) {
               window.alert('You have dragged too many files into the app. Drag just
1 file');
          } else {
               interceptArea.style.display = 'none';
               displayIconsSet();
               var file = event.dataTransfer.files[0];
               displayImageInIconSet(file.path);
          }
          return false;
     };
}

window.onload = function () {
     interceptDroppedFile();
};

109:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\ico
nic-nwjs\index.html
<html>
     <head>
          <title>Iconic</title>
          <link rel="stylesheet" href="app.css" />
          <script src="app.js"></script>
     </head>
     <body>
          <div id="load-icon-holder">
               <h1>Drag and Drop your file here</h1>
               <img src="images/drop-here.png" />
          </div>
          <div id="icons">
               <div class="icon-holder">
                    <label>16x16</label>
                    <img class="icon sixteen" />
               </div>
               <div class="icon-holder">
                    <label>32x32</label>
                    <img class="icon thirtytwo" />
               </div>
               <div class="icon-holder">
                    <label>64x64</label>
                    <img class="icon sixtyfour" />
```

```
                </div>
                <div class="icon-holder">
                    <label>128x128</label>
                    <img class="icon onetwoeight" />
                </div>
                <div class="icon-holder">
                    <label>256x256</label>
                    <img class="icon twofivesix" />
                </div>
                <div id="save">
                    <p>Click on an image to save it to your computer</p>
                </div>
            </div>
        </body>
</html>
```

110:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-nwjs\package.json

```
{
    "name":"iconic",
    "version":"1.0.0",
    "main":"index.html",
    "window": {
        "toolbar": false,
        "width": 650,
        "height": 510
    }
}
```

111:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-10\iconic-nwjs\README.md

```
# iconic
An app for converting images into different icons
```

112:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\facebomb-electron\app.js

```
'use strict';

const electron = require('electron');
const dialog = electron.remote.dialog;
const fs = require('fs');
let photoData;
let video;

function savePhoto (filePath) {
  if (filePath) {
    fs.writeFile(filePath, photoData, 'base64', (err) => {
      if (err) alert(`There was a problem saving the photo: ${err.message}`);
      photoData = null;
    });
  }
}

function initialize () {
  video = window.document.querySelector('video');
  let errorCallback = (error) => {
    console.log(`There   was   an   error   connecting   to   the   video   stream:
${error.message}`);
  };

  window.navigator.webkitGetUserMedia({video: true}, (localMediaStream) => {
    video.src = window.URL.createObjectURL(localMediaStream);
  }, errorCallback);
}

function takePhoto () {
  let canvas = window.document.querySelector('canvas');
  canvas.getContext('2d').drawImage(video, 0, 0, 800, 600);
  photoData                                                                    =
canvas.toDataURL('image/png').replace(/^data:image\/(png|jpg|jpeg);base64,/, '');
  dialog.showSaveDialog({
    title: "Save the photo",
    defaultPath: 'myfacebomb.png',
    buttonLabel: 'Save photo'
  }, savePhoto);
}

window.onload = initialize;

113:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\fac
ebomb-electron\index.html
<html>
  <head>
    <title>Facebomb</title>
    <link href="app.css" rel="stylesheet" />
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <script src="app.js"></script>
  </head>
  <body>
      <canvas width="800" height="600"></canvas>
      <video autoplay></video>
      <div id="takePhoto" onclick="takePhoto()">
     <i class="fa fa-camera" aria-hidden="true"></i>
    </div>
  </body>
</html>

114:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\fac
ebomb-electron\main.js
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    useContentSize: true,
    width: 800,
    height: 600,
    resizable: false,
    fullscreen: false
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});

115:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\fac
ebomb-electron\package.json
{
  "name": "facebomb-electron",
  "version": "1.0.0",
  "description": "An app for selfies",
  "main": "main.js",
  "scripts": {
    "start": "node_modules/.bin/electron .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "Electron"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "electron-prebuilt": "^1.2.3"
```

~~~~
  }
}

116:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\fac
ebomb-electron\README.md
# Facebomb (Electron)

An app for taking desktop selfies, built with Electron for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

![Facebomb                             Electron                             Windows
10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applicati
ons/master/app-screenshots/chapter-08/facebomb-electron-windows.png)

### Dependencies

- Node.js (4.x and above)
- Electron (1.2.1 and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This    application    was    created    for    [Cross    Platform    Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

117:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\fac
ebomb-nwjs\app.js
'use strict';

const fs = require('fs');
~~~~

```
let photoData;
let saveFile;
let video;

function bindSavingPhoto () {
  saveFile.addEventListener('change', function () {
    let filePath = this.value;
    fs.writeFile(filePath, photoData, 'base64', (err) => {
      if (err) alert('There was a problem saving the photo:', err.message);
      photoData = null;
    });
  });
}

function initialize () {
  saveFile = window.document.querySelector('#saveFile');
  video = window.document.querySelector('video');

  let errorCallback = (error) => {
    console.log('There was an error connecting to the video stream:', error);
  };

  window.navigator.webkitGetUserMedia({video: true}, (localMediaStream) => {
    video.src = window.URL.createObjectURL(localMediaStream);
    video.onloadedmetadata = bindSavingPhoto;
  }, errorCallback);
}

function takePhoto () {
  let canvas = window.document.querySelector('canvas');
  canvas.getContext('2d').drawImage(video, 0, 0, 800, 600);
  photoData                                                                    =
canvas.toDataURL('image/png').replace(/^data:image\/(png|jpg|jpeg);base64,/, '');
  saveFile.click();
}

window.onload = initialize;
```

118:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\facebomb-nwjs\index.html

```
<html>
  <head>
    <title>Facebomb</title>
    <link href="app.css" rel="stylesheet" />
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <script src="app.js"></script>
  </head>
  <body>
       <input type="file" nwsaveas="myfacebomb.png" id="saveFile">
       <canvas width="800" height="600"></canvas>
       <video autoplay></video>
       <div id="takePhoto" onclick="takePhoto()">
      <i class="fa fa-camera" aria-hidden="true"></i>
    </div>
  </body>
</html>
```

119:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\facebomb-nwjs\package.json

```
{
  "name": "facebomb",
  "version": "1.0.0",
  "main": "index.html",
  "window": {
    "toolbar": false,
    "width": 800,
    "height": 600,
    "resizable": false,
    "fullscreen": false
  },
  "dependencies": {
    "nw": "^0.15.2"
  },
  "scripts": {
    "start": "node_modules/.bin/nw ."
  }
}
```

120:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-11\facebomb-nwjs\README.md

```
# Facebomb (NW.js)

An app for taking desktop selfies, built with NW.js for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Facebomb NW.js Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/facebomb-nwjs-windows.png)

### Dependencies

- Node.js (4.x and above)
- NW.js (0.15.x and above)
```

~~~~
### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

121:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-electron\app.js
'use strict';

const electron = require('electron');
const app = electron.remote.app;

function initialize () {
     let notes = window.localStorage.notes;
     if (!notes) notes = 'Let me remember...';
     window.document.querySelector('textarea').value = notes;
}

function saveNotes () {
     let notes = window.document.querySelector('textarea').value;
     window.localStorage.setItem('notes',notes);
}

function quit () { app.quit(); }

window.onload = initialize;

122:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-electron\index.html
~~~~

```
<html>
    <head>
        <title>Let Me Remember</title>
        <link rel="stylesheet" type="text/css" href="app.css">
        <script src="app.js"></script>
    </head>
    <body>
        <div id="close" onclick="quit();">x</div>
        <textarea onKeyUp="saveNotes();"></textarea>
    </body>
</html>
```

123:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let-me-remember-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 480,
    height: 320,
    frame: false
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

124:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let-me-remember-electron\package.json

```
{
  "name": "let-me-remember-electron",
  "version": "1.0.0",
  "description": "A post-it note app for Electron",
  "main": "main.js",
  "scripts": {
    "start": "node_modules/.bin/electron .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "electron"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "electron-prebuilt": "^1.2.5"
  }
}
```

```
125:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-electron\README.md
# Let Me Remember (Electron)

A simple post-it note app, built with Electron for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

![Let me remember Electron Windows
10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applicati
ons/master/app-screenshots/chapter-08/let-me-remember-electron-windows.png)

### Dependencies

- Node.js (4.x and above)
- Electron (1.2.x and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits
```

```
&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

126:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-nwjs\app.js
'use strict';

function initialize () {
     let notes = window.localStorage.notes;
     if (!notes) notes = 'Let me remember...';
     window.document.querySelector('textarea').value = notes;
}

function saveNotes () {
     let notes = window.document.querySelector('textarea').value;
     window.localStorage.setItem('notes',notes);
}

window.onload = initialize;

127:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-nwjs\index.html
<html>
     <head>
          <title>Let Me Remember</title>
          <link rel="stylesheet" type="text/css" href="app.css">
          <script src="app.js"></script>
     </head>
     <body>
          <div id="close" onclick="process.exit(0)">x</div>
          <textarea onKeyUp="saveNotes();"></textarea>
     </body>
</html>

128:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let
-me-remember-nwjs\package.json
{
  "name": "let-me-remember",
  "version": "1.0.0",
  "main": "index.html",
  "window": {
    "width": 480,
    "height": 320,
    "frame": false,
    "toolbar": false
  },
  "scripts": {
    "start": "node_modules/.bin/nw ."
  },
  "dependencies": {
    "nw": "^0.15.3"
  }
}
```

129:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\let-me-remember-nwjs\README.md

~~~~
# Let Me Remember (NW.js)

A simple post-it note app, built with NW.js for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Let me remember NW.js Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/let-me-remember-nwjs-windows.png)

### Dependencies

- Node.js (4.x and above)
- NW.js (0.15.x and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.
~~~~

130:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-electron\index.html

```
<!doctype html>
<html lang="en" data-framework="react">
     <head>
          <meta charset="utf-8">
          <title>React 鈥?TodoMVC</title>
          <link rel="stylesheet" href="node_modules/todomvc-common/base.css">
          <link rel="stylesheet" href="node_modules/todomvc-app-css/index.css">
     </head>
     <body>
          <section class="todoapp"></section>
          <footer class="info">
               <p>Double-click to edit a todo</p>
               <p>Created by <a href="http://github.com/petehunt/">petehunt</a></p>
               <p>Part of <a href="http://todomvc.com">TodoMVC</a></p>
          </footer>

          <script src="node_modules/todomvc-common/base.js"></script>
          <script src="node_modules/react/dist/react-with-addons.js"></script>
          <script src="node_modules/classnames/index.js"></script>
          <script src="node_modules/react/dist/JSXTransformer.js"></script>
          <script src="node_modules/director/build/director.js"></script>

          <script src="js/utils.js"></script>
          <script src="js/todoModel.js"></script>
          <!-- jsx is an optional syntactic sugar that transforms methods in React's
          `render` into an HTML-looking format. Since the two models above are
          unrelated to React, we didn't need those transforms. -->
          <script type="text/jsx" src="js/todoItem.jsx"></script>
          <script type="text/jsx" src="js/footer.jsx"></script>
          <script type="text/jsx" src="js/app.jsx"></script>
     </body>
</html>
```

131:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-electron\js\todoModel.js

```
/*jshint quotmark:false */
/*jshint white:false */
/*jshint trailing:false */
/*jshint newcap:false */
var app = app || {};

(function () {
     'use strict';

     var Utils = app.Utils;
     // Generic "model" object. You can use whatever
     // framework you want. For this application it
     // may not even be worth separating this logic
     // out, but we do this to demonstrate one way to
     // separate out parts of your application.
     app.TodoModel = function (key) {
          this.key = key;
          this.todos = Utils.store(key);
          this.onChanges = [];
     };

     app.TodoModel.prototype.subscribe = function (onChange) {
          this.onChanges.push(onChange);
     };

     app.TodoModel.prototype.inform = function () {
          Utils.store(this.key, this.todos);
          this.onChanges.forEach(function (cb) { cb(); });
     };

     app.TodoModel.prototype.addTodo = function (title) {
          this.todos = this.todos.concat({
               id: Utils.uuid(),
               title: title,
               completed: false
          });

          this.inform();
     };

     app.TodoModel.prototype.toggleAll = function (checked) {
          // Note: it's usually better to use immutable data structures since they're
          // easier to reason about and React works very well with them. That's why
          // we use map() and filter() everywhere instead of mutating the array or
          // todo items themselves.
          this.todos = this.todos.map(function (todo) {
               return Utils.extend({}, todo, {completed: checked});
          });

          this.inform();
     };

     app.TodoModel.prototype.toggle = function (todoToToggle) {
          this.todos = this.todos.map(function (todo) {
               return todo !== todoToToggle ?
                   todo :
                   Utils.extend({}, todo, {completed: !todo.completed});
          });
```

```
			this.inform();
		};

		app.TodoModel.prototype.destroy = function (todo) {
			this.todos = this.todos.filter(function (candidate) {
				return candidate !== todo;
			});

			this.inform();
		};

		app.TodoModel.prototype.save = function (todoToSave, text) {
			this.todos = this.todos.map(function (todo) {
				return todo !== todoToSave ? todo : Utils.extend({}, todo, {title:
text});
			});

			this.inform();
		};

		app.TodoModel.prototype.clearCompleted = function () {
			this.todos = this.todos.filter(function (todo) {
				return !todo.completed;
			});

			this.inform();
		};

})();

132:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\tod
omvc-react-electron\js\utils.js
var app = app || {};

(function () {
	'use strict';

	app.Utils = {
		uuid: function () {
			/*jshint bitwise:false */
			var i, random;
			var uuid = '';

			for (i = 0; i < 32; i++) {
				random = Math.random() * 16 | 0;
				if (i === 8 || i === 12 || i === 16 || i === 20) {
					uuid += '-';
```

```
                }
                uuid += (i === 12 ? 4 : (i === 16 ? (random & 3 | 8) : random))
                    .toString(16);
            }

            return uuid;
        },

        pluralize: function (count, word) {
            return count === 1 ? word : word + 's';
        },

        store: function (namespace, data) {
            if (data) {
                return localStorage.setItem(namespace, JSON.stringify(data));
            }

            var store = localStorage.getItem(namespace);
            return (store && JSON.parse(store)) || [];
        },

        extend: function () {
            var newObj = {};
            for (var i = 0; i < arguments.length; i++) {
                var obj = arguments[i];
                for (var key in obj) {
                    if (obj.hasOwnProperty(key)) {
                        newObj[key] = obj[key];
                    }
                }
            }
            return newObj;
        }
    };
})();
```

133:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

134:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-electron\package.json

```
{
  "private": true,
  "dependencies": {
    "classnames": "^2.1.5",
    "director": "^1.2.0",
    "react": "^0.13.3",
    "todomvc-app-css": "^2.0.0",
    "todomvc-common": "^1.0.1"
  },
     "main":"main.js"
}
```

135:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-electron\readme.md

```
# React TodoMVC Example

> React is a JavaScript library for creating user interfaces. Its core principles are declarative code, efficiency, and flexibility. Simply specify what your component looks like and React will keep it up-to-date when the underlying data changes.

> _[React - facebook.github.io/react](http://facebook.github.io/react)_


## Learning React

The [React getting started documentation](http://facebook.github.io/react/docs/getting-started.html) is a great way to get started.

Here are some links you may find helpful:

* [Documentation](http://facebook.github.io/react/docs/getting-started.html)
* [API Reference](http://facebook.github.io/react/docs/reference.html)
* [Blog](http://facebook.github.io/react/blog/)
* [React on GitHub](https://github.com/facebook/react)
* [Support](http://facebook.github.io/react/support.html)
```

```
Articles and guides from the community:

* [How is Facebook's React JavaScript library](http://www.quora.com/React-JS-Library/How-is-Facebooks-React-JavaScript-library)
* [React: Under the hood](http://www.quora.com/Pete-Hunt/Posts/React-Under-the-Hood)

Get help from other React users:

* [React on StackOverflow](http://stackoverflow.com/questions/tagged/reactjs)
* [Discussion Forum](https://discuss.reactjs.org/)

_If you have other helpful links to share, or find any of the links above no longer work, please [let us know](https://github.com/tastejs/todomvc/issues)._


## Running

The app is built with [JSX](http://facebook.github.io/react/docs/jsx-in-depth.html) and compiled at runtime for a lighter and more fun code reading experience. As stated in the link, JSX is not mandatory.

To run the app, spin up an HTTP server (e.g. `python -m SimpleHTTPServer`) and visit http://localhost/.../myexample/.

136:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-nwjs\index.html
<!doctype html>
<html lang="en" data-framework="react">
    <head>
        <meta charset="utf-8">
        <title>React 鈥?TodoMVC</title>
        <link rel="stylesheet" href="node_modules/todomvc-common/base.css">
        <link rel="stylesheet" href="node_modules/todomvc-app-css/index.css">
    </head>
    <body>
        <section class="todoapp"></section>
        <footer class="info">
            <p>Double-click to edit a todo</p>
            <p>Created by <a href="http://github.com/petehunt/">petehunt</a></p>
            <p>Part of <a href="http://todomvc.com">TodoMVC</a></p>
        </footer>

        <script src="node_modules/todomvc-common/base.js"></script>
        <script src="node_modules/react/dist/react-with-addons.js"></script>
        <script src="node_modules/classnames/index.js"></script>
        <script src="node_modules/react/dist/JSXTransformer.js"></script>
        <script src="node_modules/director/build/director.js"></script>

        <script src="js/utils.js"></script>
        <script src="js/todoModel.js"></script>
        <!-- jsx is an optional syntactic sugar that transforms methods in React's
        `render` into an HTML-looking format. Since the two models above are
        unrelated to React, we didn't need those transforms. -->
        <script type="text/jsx" src="js/todoItem.jsx"></script>
        <script type="text/jsx" src="js/footer.jsx"></script>
        <script type="text/jsx" src="js/app.jsx"></script>
    </body>
</html>
```

137:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-nwjs\js\todoModel.js

```
/*jshint quotmark:false */
/*jshint white:false */
/*jshint trailing:false */
/*jshint newcap:false */
var app = app || {};

(function () {
    'use strict';

    var Utils = app.Utils;
    // Generic "model" object. You can use whatever
    // framework you want. For this application it
    // may not even be worth separating this logic
    // out, but we do this to demonstrate one way to
    // separate out parts of your application.
    app.TodoModel = function (key) {
        this.key = key;
        this.todos = Utils.store(key);
        this.onChanges = [];
    };

    app.TodoModel.prototype.subscribe = function (onChange) {
        this.onChanges.push(onChange);
    };

    app.TodoModel.prototype.inform = function () {
        Utils.store(this.key, this.todos);
        this.onChanges.forEach(function (cb) { cb(); });
    };

    app.TodoModel.prototype.addTodo = function (title) {
```

```
		this.todos = this.todos.concat({
			id: Utils.uuid(),
			title: title,
			completed: false
		});

		this.inform();
	};

	app.TodoModel.prototype.toggleAll = function (checked) {
		// Note: it's usually better to use immutable data structures since they're
		// easier to reason about and React works very well with them. That's why
		// we use map() and filter() everywhere instead of mutating the array or
		// todo items themselves.
		this.todos = this.todos.map(function (todo) {
			return Utils.extend({}, todo, {completed: checked});
		});

		this.inform();
	};

	app.TodoModel.prototype.toggle = function (todoToToggle) {
		this.todos = this.todos.map(function (todo) {
			return todo !== todoToToggle ?
				todo :
				Utils.extend({}, todo, {completed: !todo.completed});
		});

		this.inform();
	};

	app.TodoModel.prototype.destroy = function (todo) {
		this.todos = this.todos.filter(function (candidate) {
			return candidate !== todo;
		});

		this.inform();
	};

	app.TodoModel.prototype.save = function (todoToSave, text) {
		this.todos = this.todos.map(function (todo) {
			return todo !== todoToSave ? todo : Utils.extend({}, todo, {title: 
text});
		});

		this.inform();
	};
	app.TodoModel.prototype.clearCompleted = function () {
		this.todos = this.todos.filter(function (todo) {
			return !todo.completed;
		});

		this.inform();
	};

})();

138:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\tod
omvc-react-nwjs\js\utils.js
var app = app || {};

(function () {
	'use strict';

	app.Utils = {
		uuid: function () {
			/*jshint bitwise:false */
			var i, random;
			var uuid = '';

			for (i = 0; i < 32; i++) {
				random = Math.random() * 16 | 0;
				if (i === 8 || i === 12 || i === 16 || i === 20) {
					uuid += '-';
				}
				uuid += (i === 12 ? 4 : (i === 16 ? (random & 3 | 8) : random))
					.toString(16);
			}

			return uuid;
		},

		pluralize: function (count, word) {
			return count === 1 ? word : word + 's';
		},

		store: function (namespace, data) {
			if (data) {
				return localStorage.setItem(namespace, JSON.stringify(data));
			}

			var store = localStorage.getItem(namespace);
			return (store && JSON.parse(store)) || [];
```

```
        },

        extend: function () {
            var newObj = {};
            for (var i = 0; i < arguments.length; i++) {
                var obj = arguments[i];
                for (var key in obj) {
                    if (obj.hasOwnProperty(key)) {
                        newObj[key] = obj[key];
                    }
                }
            }
            return newObj;
        }
    };
})();
```

139:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-nwjs\package.json

```
{
    "name":"todo-mvc-app",
    "version":"1.0.0",
    "main":"index.html",
    "window": {
        "toolbar":false
    },
  "private": true,
  "dependencies": {
    "classnames": "^2.1.5",
    "director": "^1.2.0",
    "react": "^0.13.3",
    "todomvc-app-css": "^2.0.0",
    "todomvc-common": "^1.0.1"
  }
}
```

140:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-12\todomvc-react-nwjs\readme.md

```
# React TodoMVC Example

> React is a JavaScript library for creating user interfaces. Its core principles are declarative code, efficiency, and flexibility. Simply specify what your component looks like and React will keep it up-to-date when the underlying data changes.

> _[React - facebook.github.io/react](http://facebook.github.io/react)_
```

## Learning React

The [React getting started documentation](http://facebook.github.io/react/docs/getting-started.html) is a great way to get started.

Here are some links you may find helpful:

* [Documentation](http://facebook.github.io/react/docs/getting-started.html)
* [API Reference](http://facebook.github.io/react/docs/reference.html)
* [Blog](http://facebook.github.io/react/blog/)
* [React on GitHub](https://github.com/facebook/react)
* [Support](http://facebook.github.io/react/support.html)

Articles and guides from the community:

* [How is Facebook's React JavaScript library](http://www.quora.com/React-JS-Library/How-is-Facebooks-React-JavaScript-library)
* [React: Under the hood](http://www.quora.com/Pete-Hunt/Posts/React-Under-the-Hood)

Get help from other React users:

* [React on StackOverflow](http://stackoverflow.com/questions/tagged/reactjs)
* [Discussion Forum](https://discuss.reactjs.org/)

If you have other helpful links to share, or find any of the links above no longer work, please [let us know](https://github.com/tastejs/todomvc/issues).

## Running

The app is built with [JSX](http://facebook.github.io/react/docs/jsx-in-depth.html) and compiled at runtime for a lighter and more fun code reading experience. As stated in the link, JSX is not mandatory.

To run the app, spin up an HTTP server (e.g. `python -m SimpleHTTPServer`) and visit http://localhost/.../myexample/.

141:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\app.js

```
'use strict';

const electron = require('electron');
const clipboard = electron.clipboard;
const phrases = require('./phrases');
let phrasesArea;
let template;

function addPhrase (phrase) {
  template.content.querySelector('div').innerText = phrase;
  let clone = window.document.importNode(template.content, true);
  phrasesArea.appendChild(clone);
}

function loadPhrasesIntoApp () {
  phrasesArea = window.document.getElementById('phrases');
  template = window.document.querySelector('#phrase');
  phrases.forEach(addPhrase);
}

function copyPhraseToClipboard (phrase) {
  clipboard.writeText(phrase);
}

window.onload = loadPhrasesIntoApp;
```

142:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\index.html

```
<html>
  <head>
    <title>Pearls</title>
    <link href="app.css" rel="stylesheet" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="phrase">
      <div class="phrase" onclick="copyPhraseToClipboard(this.innerText);"></div>
    </template>
    <div id="phrases"></div>
  </body>
</html>
```

143:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 670,
    height: 550,
    useContentSize: true
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

144:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\package.json

```
{
  "name": "pearls-electron",
  "version": "1.0.0",
  "description": "A clipboard API example for Electron and the book 'Cross Platform Desktop Applications'",
  "main": "main.js",
  "scripts": {
    "start": "node_modules/.bin/electron .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "electron",
    "clipboard"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "electron": "^1.3.7"
  }
}
```

145:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\phrases.js

```
'use strict';

module.exports = [
     'I have to return some videotapes',
     'Do not attempt to grow a brain',
     'So tell me, do you feel lucky? Well do ya, Punk!',
     'We\'re gonna need a bigger boat',
     'We can handle a little chop',
     'Get to the choppa!',
```

~~~
    'Hold onto your butts',
    'Today we\'re going to play a wonderful game called "Who is your daddy, and what does he do?"',
    'Yesterday we were an army without a country. Tomorrow we must decide... which country we want to buy!'
];

146:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-electron\README.md
# Pearls (Electron)

A quotes app, built with Electron for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Pearls Electron Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/pearls-electron-windows.png)

### Dependencies

- Node.js (4.x and above)
- Electron (1.2.4 and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

147:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pea
~~~

rls-nwjs\app.js

```
'use strict';

const gui = require('nw.gui');
const clipboard = gui.Clipboard.get();
const phrases = require('./phrases');
let phrasesArea;
let template;

function addPhrase (phrase) {
  template.content.querySelector('div').innerText = phrase;
  let clone = window.document.importNode(template.content, true);
  phrasesArea.appendChild(clone);
}

function loadPhrasesIntoApp () {
  phrasesArea = window.document.getElementById('phrases');
  template = window.document.querySelector('#phrase');
  phrases.forEach(addPhrase);
}

function copyPhraseToClipboard (phrase) {
  clipboard.set(phrase, 'text');
}

window.onload = loadPhrasesIntoApp;
```

148:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-nwjs\index.html

```
<html>
  <head>
    <title>Pearls</title>
    <link href="app.css" rel="stylesheet" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="phrase">
      <div class="phrase" onclick="copyPhraseToClipboard(this.innerText);"></div>
    </template>
    <div id="phrases"></div>
  </body>
</html>
```

149:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-nwjs\package.json

```
{
  "name":"pearls",
  "version":"1.0.0",
  "main":"index.html",
  "window": {
    "width": 650,
    "height": 550,
    "toolbar": false
  },
  "scripts": {
    "start": "node_modules/.bin/nw ."
  },
  "dependencies": {
    "nw": "^0.15.3"
  }
}
```

150:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-nwjs\phrases.js

```
'use strict';

module.exports = [
     'I have to return some videotapes',
     'Do not attempt to grow a brain',
     'So tell me, do you feel lucky? Well do ya, Punk!',
     'We\'re gonna need a bigger boat',
     'We can handle a little chop',
     'Get to the choppa!',
     'Hold onto your butts',
     'Today we\'re going to play a wonderful game called "Who is your daddy, and what does he do?"',
     'Yesterday we were an army without a country. Tomorrow we must decide... which country we want to buy!'
];
```

151:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-13\pearls-nwjs\README.md

```
# Pearls (NW.js)

A quotes app, built with NW.js for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Pearls NW.js Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/pearls-nwjs-windows.png)

### Dependencies

- Node.js (4.x and above)
```

~~~
- NW.js (0.15.x and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

152:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\sna
ke-electron\app.js
'use strict';
let currentState;
let canvas, ctx, gridSize, currentPosition, snakeBody, snakeLength, direction, score,
suggestedPoint, allowPressKeys, interval, choice;

function updateScore () {
  score = (snakeLength - 3) * 10;
  document.getElementById('score').innerText = score;
}

function hasPoint (element) {
  return (element[0] === suggestedPoint[0] && element[1] === suggestedPoint[1]);
}

function makeFoodItem () {
  suggestedPoint = [Math.floor(Math.random()*(canvas.width/gridSize))*gridSize,
Math.floor(Math.random()*(canvas.height/gridSize))*gridSize];
  if (snakeBody.some(hasPoint)) {
    makeFoodItem();
  } else {
~~~

```
    ctx.fillStyle = 'rgb(10,100,0)';
    ctx.fillRect(suggestedPoint[0], suggestedPoint[1], gridSize, gridSize);
  }
}

function hasEatenItself (element) {
  return (element[0] === currentPosition.x && element[1] === currentPosition.y);
}

function leftPosition(){
 return currentPosition.x - gridSize;
}

function rightPosition(){
  return currentPosition.x + gridSize;
}

function upPosition(){
  return currentPosition.y - gridSize;
}

function downPosition(){
  return currentPosition.y + gridSize;
}

function whichWayToGo (axisType) {
  if (axisType === 'x') {
    choice = (currentPosition.x > canvas.width / 2) ? moveLeft() : moveRight();
  } else {
    choice = (currentPosition.y > canvas.height / 2) ? moveUp() : moveDown();
  }
}

function moveUp(){
  if (upPosition() >= 0) {
    executeMove('up', 'y', upPosition());
  } else {
    whichWayToGo('x');
  }
}

function moveDown(){
  if (downPosition() < canvas.height) {
    executeMove('down', 'y', downPosition());
  } else {
    whichWayToGo('x');
  }
```

```
}

function moveLeft(){
  if (leftPosition() >= 0) {
    executeMove('left', 'x', leftPosition());
     } else {
    whichWayToGo('y');
  }
}

function moveRight(){
  if (rightPosition() < canvas.width) {
    executeMove('right', 'x', rightPosition());
  } else {
    whichWayToGo('y');
  }
}

function executeMove(dirValue, axisType, axisValue) {
  direction = dirValue;
  currentPosition[axisType] = axisValue;
  drawSnake();
}

function moveSnake(){
  switch (direction) {
    case 'up':
      moveUp();
      break;

    case 'down':
      moveDown();
      break;

    case 'left':
      moveLeft();
      break;

    case 'right':
      moveRight();
      break;
  }
}

function restart () {
     document.getElementById('play_menu').style.display='block';
     document.getElementById('pause_menu').style.display='none';
     document.getElementById('restart_menu').style.display='none';
     pause();
     start();
}

function pause(){
     document.getElementById('play_menu').style.display='none';
     document.getElementById('pause_menu').style.display='block';
  clearInterval(interval);
  allowPressKeys = false;
}

function play(){
     document.getElementById('play_menu').style.display='block';
     document.getElementById('pause_menu').style.display='none';
  interval = setInterval(moveSnake,100);
  allowPressKeys = true;
}

function gameOver(){
  pause();
  window.alert('Game Over. Your score was ' + score);
  ctx.clearRect(0,0, canvas.width, canvas.height);
     document.getElementById('play_menu').style.display='none';
  document.getElementById('restart_menu').style.display='block';
}

function drawSnake() {
  if (snakeBody.some(hasEatenItself)) {
    gameOver();
    return false;
  }
  snakeBody.push([currentPosition.x, currentPosition.y]);
  ctx.fillStyle = 'rgb(200,0,0)';
  ctx.fillRect(currentPosition.x, currentPosition.y, gridSize, gridSize);
  if (snakeBody.length > snakeLength) {
    let itemToRemove = snakeBody.shift();
    ctx.clearRect(itemToRemove[0], itemToRemove[1], gridSize, gridSize);
  }
  if   (currentPosition.x   ===   suggestedPoint[0]   &&   currentPosition.y   ===
suggestedPoint[1]) {
    makeFoodItem();
    snakeLength += 1;
    updateScore();
  }
}
```

```
window.document.onkeydown = function(event) {
  if (!allowPressKeys){
    return null;
  }
  let keyCode;
  if(!event)
  {
    keyCode = window.event.keyCode;
  }
  else
  {
    keyCode = event.keyCode;
  }

  switch(keyCode)
  {
    case 37:
      if (direction !== 'right') {
        moveLeft();
      }
      break;

    case 38:
      if (direction !== 'down'){
        moveUp();
      }
      break;

    case 39:
      if (direction !== 'left'){
        moveRight();
      }
      break;

    case 40:
      if (direction !== 'up'){
        moveDown();
      }
      break;

    default:
      break;
  }
};

function start () {
  ctx.clearRect(0,0, canvas.width, canvas.height);
  currentPosition = {'x':50, 'y':50};
  snakeBody = [];
  snakeLength = 3;
  updateScore();
  makeFoodItem();
  drawSnake();
  direction = 'right';
  play();
}

function initialize () {
  canvas = document.querySelector('canvas');
  ctx = canvas.getContext('2d');
  gridSize = 10;
  start();
}

function togglePauseState () {
  if (currentState) {
    if (currentState === 'play') {
      pause();
          currentState = 'pause';
    } else {
      play();
          currentState = 'play';
    }
   } else {
     pause();
     currentState = 'play';
   }
}

const ipcRenderer = require('electron').ipcRenderer;

function togglePauseState () {
  if (currentState) {
    if (currentState === 'play') {
      pause();
      currentState = 'pause';
    } else {
      play();
      currentState = 'play';
    }
  } else {
    pause();
    currentState = 'play';
     }
```

```
}

ipcRenderer.on('togglePauseState', togglePauseState);

window.onload = initialize;
```

153:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-electron\index.html

```
<html>
  <head>
    <title>Snake</title>
    <link href="app.css" rel="stylesheet" />
    <script src="app.js"></script>
  </head>
  <body>
    <div id="scoreboard">
      <span id="label">Score:</span>
      <span id="score"></span>
              <div id="bar">
                       <div id="play_menu">
                       <button onclick="pause();">Pause</button>
                       </div>
                   <div id="pause_menu">
                     <button onclick="play();">Resume</button>
                          <button onclick="restart();">Restart</button>
                   </div>
                   <div id="restart_menu">
                     <button onclick="restart();">Restart</button>
                   </div>
                 </div>
              </div>
    </div>
    <canvas></canvas>
  </body>
</html>
```

154:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-electron\main.js

```
'use strict';

const {app, globalShortcut, BrowserWindow} = require('electron');

let mainWindow = null;

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

```
app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 840,
    height: 470,
    useContentSize: true
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
  const pauseKey = globalShortcut.register('CommandOrControl+P', () => {
    mainWindow.webContents.send('togglePauseState');
  });
  if (!pauseKey) alert('You will not be able to pause the game from the keyboard');
});

app.on('will-quit', () => {
  globalShortcut.unregister('CommandOrControl+P');
});
```

155:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-electron\package.json

```
{
  "name": "snake-electron",
  "version": "1.0.0",
  "description": "The Snake game, built with Electron for the book 'Cross Platform
Desktop Applications'",
  "main": "main.js",
  "scripts": {
    "start": "node_modules/.bin/electron .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "electron",
    "keyboard",
    "shortcuts"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "electron-prebuilt": "^1.2.5"
  }
}
```

156:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-electron\README.md

```
# Snake (Electron)
```

~~~
The Snake game, built with Electron for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Snake Electron Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/snake-electron-windows.png)

### Dependencies

- Node.js (4.x and above)
- Electron (1.2.5 and above)

### Installation

```
cd PATH_TO_THIS_APP
npm install
```

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

157:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-nwjs\app.js
'use strict';
let currentState;
let canvas, ctx, gridSize, currentPosition, snakeBody, snakeLength, direction, score, suggestedPoint, allowPressKeys, interval, choice;

function updateScore () {
  score = (snakeLength - 3) * 10;
  document.getElementById('score').innerText = score;
}
~~~

```
function hasPoint (element) {
  return (element[0] === suggestedPoint[0] && element[1] === suggestedPoint[1]);
}

function makeFoodItem () {
  suggestedPoint   =   [Math.floor(Math.random()*(canvas.width/gridSize))*gridSize,
Math.floor(Math.random()*(canvas.height/gridSize))*gridSize];
  if (snakeBody.some(hasPoint)) {
    makeFoodItem();
  } else {
    ctx.fillStyle = 'rgb(10,100,0)';
    ctx.fillRect(suggestedPoint[0], suggestedPoint[1], gridSize, gridSize);
  }
}

function hasEatenItself (element) {
  return (element[0] === currentPosition.x && element[1] === currentPosition.y);
}

function leftPosition(){
 return currentPosition.x - gridSize;
}

function rightPosition(){
  return currentPosition.x + gridSize;
}

function upPosition(){
  return currentPosition.y - gridSize;
}

function downPosition(){
  return currentPosition.y + gridSize;
}

function whichWayToGo (axisType) {
  if (axisType === 'x') {
    choice = (currentPosition.x > canvas.width / 2) ? moveLeft() : moveRight();
  } else {
    choice = (currentPosition.y > canvas.height / 2) ? moveUp() : moveDown();
  }
}

function moveUp(){
  if (upPosition() >= 0) {
    executeMove('up', 'y', upPosition());
  } else {
    whichWayToGo('x');
  }
}

function moveDown(){
  if (downPosition() < canvas.height) {
    executeMove('down', 'y', downPosition());
  } else {
    whichWayToGo('x');
  }
}

function moveLeft(){
  if (leftPosition() >= 0) {
    executeMove('left', 'x', leftPosition());
     } else {
    whichWayToGo('y');
  }
}

function moveRight(){
  if (rightPosition() < canvas.width) {
    executeMove('right', 'x', rightPosition());
  } else {
    whichWayToGo('y');
  }
}

function executeMove(dirValue, axisType, axisValue) {
  direction = dirValue;
  currentPosition[axisType] = axisValue;
  drawSnake();
}

function moveSnake(){
  switch (direction) {
    case 'up':
      moveUp();
      break;

    case 'down':
      moveDown();
      break;

    case 'left':
      moveLeft();
      break;
    case 'right':
      moveRight();
      break;
  }
}

function restart () {
     document.getElementById('play_menu').style.display='block';
     document.getElementById('pause_menu').style.display='none';
     document.getElementById('restart_menu').style.display='none';
     pause();
     start();
}

function pause(){
     document.getElementById('play_menu').style.display='none';
     document.getElementById('pause_menu').style.display='block';
  clearInterval(interval);
  allowPressKeys = false;
}

function play(){
     document.getElementById('play_menu').style.display='block';
     document.getElementById('pause_menu').style.display='none';
  interval = setInterval(moveSnake,100);
  allowPressKeys = true;
}

function gameOver(){
  pause();
  window.alert('Game Over. Your score was ' + score);
  ctx.clearRect(0,0, canvas.width, canvas.height);
     document.getElementById('play_menu').style.display='none';
  document.getElementById('restart_menu').style.display='block';
}

function drawSnake() {
  if (snakeBody.some(hasEatenItself)) {
    gameOver();
    return false;
  }
  snakeBody.push([currentPosition.x, currentPosition.y]);
  ctx.fillStyle = 'rgb(200,0,0)';
  ctx.fillRect(currentPosition.x, currentPosition.y, gridSize, gridSize);
  if (snakeBody.length > snakeLength) {
    let itemToRemove = snakeBody.shift();
    ctx.clearRect(itemToRemove[0], itemToRemove[1], gridSize, gridSize);
  }
  if   (currentPosition.x   ===   suggestedPoint[0]   &&   currentPosition.y   ===
suggestedPoint[1]) {
    makeFoodItem();
    snakeLength += 1;
    updateScore();
  }
}

window.document.onkeydown = function(event) {
  if (!allowPressKeys){
    return null;
  }
  let keyCode;
  if(!event)
  {
    keyCode = window.event.keyCode;
  }
  else
  {
    keyCode = event.keyCode;
  }

  switch(keyCode)
  {
    case 37:
      if (direction !== 'right') {
        moveLeft();
      }
      break;

    case 38:
      if (direction !== 'down'){
        moveUp();
      }
      break;

    case 39:
      if (direction !== 'left'){
        moveRight();
      }
      break;

    case 40:
      if (direction !== 'up'){
        moveDown();
      }
      break;

    default:
      break;
  }
};

function start () {
  ctx.clearRect(0,0, canvas.width, canvas.height);
  currentPosition = {'x':50, 'y':50};
  snakeBody = [];
  snakeLength = 3;
  updateScore();
  makeFoodItem();
  drawSnake();
  direction = 'right';
  play();
}

function initialize () {
  canvas = document.querySelector('canvas');
  ctx = canvas.getContext('2d');
  gridSize = 10;
  start();
}

function togglePauseState () {
  if (currentState) {
    if (currentState === 'play') {
      pause();
          currentState = 'pause';
    } else {
      play();
          currentState = 'play';
    }
   } else {
     pause();
     currentState = 'play';
   }
}

const pauseKeyOptions = {
  key:'Ctrl+P',
  active: togglePauseState,
  failed: () => {
    console.log('An error occurred');
  }
};

const pauseShortcut = new nw.Shortcut(pauseKeyOptions);
nw.App.registerGlobalHotKey(pauseShortcut);
process.on('exit', () => {
  nw.App.unregisterGlobalHotKey(pauseShortcut);
});


window.onload = initialize;
```

158:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-nwjs\index.html

```
<html>
  <head>
    <title>Snake</title>
    <link href="app.css" rel="stylesheet" />
    <script src="app.js"></script>
  </head>
  <body>
    <div id="scoreboard">
      <span id="label">Score:</span>
      <span id="score"></span>
              <div id="bar">
                       <div id="play_menu">
                       <button onclick="pause();">Pause</button>
                       </div>
                     <div id="pause_menu">
                       <button onclick="play();">Resume</button>
                             <button onclick="restart();">Restart</button>
                     </div>
                     <div id="restart_menu">
                       <button onclick="restart();">Restart</button>
                     </div>
                   </div>
              </div>
    </div>
    <canvas></canvas>
  </body>
</html>
```

159:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-nwjs\package.json

```
{
  "name": "snake-nwjs",
  "version": "1.0.0",
  "description": "A Snake game in NW.js for 'Cross Platform Desktop Applications'",
  "main": "index.html",
  "scripts": {
    "start": "node_modules/.bin/nw .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "snake",
    "nwjs"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "window": {
    "width": 840,
    "height": 470,
    "toolbar": false
  },
  "dependencies": {
    "nw": "^0.15.3"
  }
}
```

160:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-14\snake-nwjs\README.md

```
# Snake (NW.js)

The Snake game, built with NW.js for [Cross Platform Desktop Applications](https://manning.com/books/cross-platform-desktop-applications).

![Snake NW.js Windows 10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applications/master/app-screenshots/chapter-08/snake-nwjs-windows.png)

### Dependencies

- Node.js (4.x and above)
- NW.js (0.15.x and above)

### Installation

` ` `
cd PATH_TO_THIS_APP
npm install
` ` `

### Starting the app
```

~~~
```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

161:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-electron\app.js
'use strict';

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

function search () {
  const formInput = window.document.querySelector('form input');
  const term = formInput.value;
  ipcRenderer.send('monitorTerm', term);
  return false;
}

162:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-electron\config.example.js
module.exports = {
     consumer_key: null,
  consumer_secret: null,
  access_token_key: null,
  access_token_secret: null
};

163:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-electron\index.html
<html>
  <head>
    <title>Watchy</title>
    <link rel="stylesheet" href="app.css"/>
    <script src="app.js"></script>
  </head>
  <body>
    <form onsubmit="search();">
      <input type="text" placeholder="Monitor tweets about..." />
~~~

```
      <button type="submit">Monitor</button>
    </form>
  </body>
</html>
```

164:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\watchy-electron\main.js

```
'use strict';

const {app, ipcMain, BrowserWindow} = require('electron');
const notifier = require('electron-notifications');
var config = require('./config');
var Twitter = require('twitter');
var client = new Twitter(config);

let mainWindow = null;

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

ipcMain.on('monitorTerm', (event, term) => {
  client.stream('statuses/filter', {track: term}, (stream) => {
       stream.on('data', (tweet) => {
      let notification = notifier.notify('New tweet', {
        icon: tweet.user.profile_image_url,
        message: tweet.text
      });
    });
          stream.on('error', (error) => {
            console.log(error.message);
       });
     });
});

app.on('ready', () => {
  mainWindow = new BrowserWindow({
    width: 370,
    height: 90,
    useContentSize: true
  });
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

165:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\watchy-electron\package.json

```
{
  "name": "watchy-electron",
  "version": "1.0.0",
  "description": "A Twitter client for monitoring topics, built with Electron for the
book 'Cross Platform Desktop Applications'",
  "main": "main.js",
  "scripts": {
    "start": "node_modules/.bin/electron .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "electron",
    "twitter"
  ],
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "electron-notifications": "0.0.3",
    "electron": "^1.3.7",
    "twitter": "^1.3.0"
  }
}
```

166:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-electron\README.md

```
# Watchy (Electron)

A Twitter client for monitoring topics, built with Electron for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

![Watchy Electron Windows
10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applicati
ons/master/app-screenshots/chapter-08/watchy-electron-windows.png)

### Dependencies

- Node.js (4.x and above)
- NW.js (0.15.x and above)

You'll also need to create a Twitter app via Twitter's developer API. For more
information, see here: https://dev.twitter.com

### Installation

```
cd PATH_TO_THIS_APP
npm install
```
```

~~~~
cp config.example.js config.js
```

After creating the config.js file, fill in the null values with the API credentials
for your Twitter application.

### Starting the app

```
cd PATH_TO_THIS_APP
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

167:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-nwjs\app.js
'use strict';

const Twitter = require('twitter');
const config  = require('./config');
var term;
const client = new Twitter(config);
let notify = Notification;

function notifyOfTweet (tweet) {
  new notify(`New tweet about ${term}`,
    {
      body: tweet.text,
      icon: tweet.user.profile_image_url
    }
  );
}

function search () {
  var formInput = window.document.querySelector('form input');
  term = formInput.value;
  client.stream('statuses/filter', {track: term}, (stream) => {
    stream.on('data', notifyOfTweet);
    stream.on('error', (error) => {
~~~~

```
      alert(error.message);
    });
  });
  return false;
}
```

168:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\watchy-nwjs\config.example.js

```
module.exports = {
     consumer_key: null,
  consumer_secret: null,
  access_token_key: null,
  access_token_secret: null
};
```

169:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\watchy-nwjs\index.html

```
<html>
     <head>
          <title>Watchy</title>
          <link rel="stylesheet" href="app.css"/>
          <script src="app.js"></script>
     </head>
     <body>
          <form onsubmit="search();">
               <input type="text" placeholder="Monitor tweets about..." />
               <button type="submit">Monitor</button>
          </form>
     </body>
</html>
```

170:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\watchy-nwjs\package.json

```
{
  "name": "watchy-nwjs",
  "version": "1.0.0",
  "description": "A Twitter client for monitoring topics, built with NW.js for the book
'Cross Platform Desktop Applications'",
  "main": "index.html",
  "scripts": {
    "start": "node_modules/.bin/nw .",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "twitter",
    "nwjs"
  ],
```

~~~~
  "window": {
    "toolbar": true,
    "width": 370,
    "height": 80
  },
  "author": "Paul Jensen <paulbjensen@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "nw": "^0.15.3",
    "twitter": "^1.3.0"
  }
}

171:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-15\wat
chy-nwjs\README.md
# Watchy (NW.js)

A Twitter client for monitoring topics, built with NW.js for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

![Watchy                                NW.js                                Windows
10](https://raw.githubusercontent.com/paulbjensen/cross-platform-desktop-applicati
ons/master/app-screenshots/chapter-08/watchy-nwjs-windows.png)

### Dependencies

- Node.js (4.x and above)
- NW.js (0.15.x and above)

You'll also need to create a Twitter app via Twitter's developer API. For more
information, see here: https://dev.twitter.com

### Installation

```
cd PATH_TO_THIS_APP
npm install
cp config.example.js config.js
```

After creating the config.js file, fill in the null values with the API credentials
for your Twitter application.

### Starting the app

```
cd PATH_TO_THIS_APP
~~~~

~~~~
npm start
```

### About this application

This application was created for [Cross Platform Desktop
Applications](https://manning.com/books/cross-platform-desktop-applications).

### Licence and Credits

&copy; 2016 Paul Jensen. The app source code is licensed under the MIT License.

172:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\app.js
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

173:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\cuke.js
'use strict';

// Dependencies
const exec = require('child_process').exec;
const path = require('path');

let command = 'node_modules/.bin/cucumber-js';
if (process.platform === 'win32') command += '.cmd';
~~~~

```
exec(path.join(process.cwd(), command), (err, stdout, stderr) => {
  console.log(stdout);
  console.log(stderr);
});

174:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\features\step_definitions\image_steps.js
'use strict';

// Dependencies
//
const assert = require('assert');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');
const {defineSupportCode} = require('cucumber');

defineSupportCode(
     function({Then, When, Given}) {

          Given(/^I have the application open and running$/, {timeout: 20 * 1000},
function (callback) {
         const self = this;

         self.app.start().then(() => {
           return self.app.browserWindow.isVisible();
         }).then((isVisible) => {
           assert.equal(isVisible, true);
           callback();
         })

       });

       When(/^I search for "([^"]*)"$/, function (term, callback) {
         this.app.client.setValue('#search', term)
         .then(() => { callback(); });
       });

       When(/^I double click on the "([^"]*)" folder$/, function (folderName, callback)
{
         const folderPath = path.join(osenv.home(),folderName);
         this.app.client.doubleClick(`//img[@data-filepath="${folderPath}"]`)
         .then(() => { callback(); });
       });

       When(/^I double click on "([^"]*)"$/, function (fileName, callback) {
         const filePath = path.join(osenv.home(),fileName);
```

```
        this.app.client.doubleClick(`//img[@data-filepath="${filePath}"]`)
        .then(() => { callback(); });
      });

       Then(/^I should see the "([^"]*)" file opened in a photo app$/, function
(fileName, callback) {
        const filePath = path.join(osenv.home(),fileName);
        setTimeout(function () {
          fs.stat(filePath, function (err, stat) {
            const timeDifference = Date.now() - stat.atime.getTime();
            assert.equal(null, err);
            assert(timeDifference < 3000);
            callback(err);
          });
        }, 3000);
      });

      When(/^I wait (\d+) seconds$/, (numberOfSeconds, callback) => {
          setTimeout(callback, numberOfSeconds * 1000);
      });

    }
);

175:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\features\support\hooks.js
'use strict';

const Application = require('spectron').Application;
const path = require('path');
let electronPath = path.join(__dirname, '../../node_modules/.bin/electron');
const entryPointPath = path.join(__dirname, '../../main.js');
if (process.platform === 'win32') electronPath += '.cmd';
const {defineSupportCode} = require('cucumber');

defineSupportCode(function ({Before, After}) {

    Before(function (scenario, callback) {
    this.app = new Application({
      path: electronPath,
      args: [entryPointPath]
    });
    callback();
  });

    After(function (scenario, callback) {
         this.app.stop();
    callback();
     });

});
```

176:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-electron\fileSystem.js

```
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}
```

```
function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

177:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-electron\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

178:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

let appPath = app.getAppPath();
if (process.env.NODE_ENV === 'test') appPath = process.cwd();

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${appPath}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

179:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-electron\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "main.js",
  "author": "Paul Jensen <paul@anephenix.com>",
  "description": "A file explorer application",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  },
  "scripts": {
        "cuke": "NODE_ENV=test node_modules/.bin/cuke",
    "test": "NODE_ENV=test node_modules/.bin/mocha",
    "pack": "build",
    "dist": "build"
  },
  "devDependencies": {
    "cucumber": "^2.0.0-rc.7",
    "electron": "^1.4.14",
    "electron-builder": "^11.4.4",
    "mocha": "^3.2.0",
    "spectron": "^3.5.0"
  },
  "build": {}
}
```

180:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor

```
ikeet-electron\search.js
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

181:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\test\folderExplorer.test.js
'use strict';

const Application = require('spectron').Application;
const assert = require('assert');
const path = require('path');
const osenv = require('osenv');

let app;
let electronPath = path.join(__dirname, '../node_modules/.bin/electron');
let entryPointPath = path.join(__dirname, '../main.js');
if (process.platform === 'win32') electronPath += '.cmd';

describe('exploring folders', () => {

  beforeEach(() => {
    return app = new Application({
      path: electronPath,
      args: [entryPointPath]
    });
  });

  it('should allow the user to navigate folders by double-clicking on them', function
(done) {

    function finish (error) {
      app.stop();
      return done(error);
    }

    let documentsFilePath = path.join(osenv.home(),'/Documents');

    this.timeout(10000);
    app.start().then(() => {
      return app.browserWindow.isVisible();
    }).then((isVisible) => {
      assert.equal(isVisible, true);
    }).then(() => {
      return
app.client.doubleClick(`//img[@data-filepath="${documentsFilePath}"]`);
    }).then(() => {
      return app.client.getText('#current-folder');
    }).then((currentFolder) => {
      assert.equal(documentsFilePath, currentFolder);
    })
    .then(finish)
    .catch(finish);
  });

});

182:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\test\search.test.js
'use strict';

const assert = require('assert');
const lunr = require('lunr');
global.window = {};
global.window.lunr = lunr;
const search = require('../search');

describe('search', () => {
  describe('#find', () => {
        it('should return results when a file matches a term', (done) => {
                const seedFileReferences = [
            {
              file: 'john.png',
              type: 'image/png',
              path: '/Users/pauljensen/Pictures/john.png'
            },
            {
              file: 'bob.png',
              type: 'image/png',
              path: '/Users/pauljensen/Pictures/bob.png'
            },
            {
              file: 'frank.png',
              type: 'image/png',
              path: '/Users/pauljensen/Pictures/frank.png'
            }
      ];

      search.resetIndex();
      seedFileReferences.forEach(search.addToIndex);

      search.find('frank', (results) => {
           assert(results.length === 1);
           assert.equal(seedFileReferences[2].path, results[0].ref);
            done();
          });
           });
  });
});

183:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-electron\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}
```

```
function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                              class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
```

```
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };

184:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-nwjs\app.js
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

185:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-nwjs\fileSystem.js
'use strict';
```

```
const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

186:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
              <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

187:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  },
  "devDependencies": {
    "mocha": "^3.2.0"
  }
}
```

```
188:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-nwjs\search.js
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

189:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lor
ikeet-nwjs\test\search.test.js
'use strict';

const assert = require('assert');
const lunr = require('lunr');
global.window = {};
global.window.lunr = lunr;
const search = require('../search');

describe('search', () => {
  describe('#find', () => {
         it('should return results when a file matches a term', (done) => {

              const seedFileReferences = [
           {
            file: 'john.png',
            type: 'image/png',
            path: '/Users/pauljensen/Pictures/john.png'
          },
          {
            file: 'bob.png',
            type: 'image/png',
            path: '/Users/pauljensen/Pictures/bob.png'
          },
          {
            file: 'frank.png',
            type: 'image/png',
            path: '/Users/pauljensen/Pictures/frank.png'
          }
      ];

      search.resetIndex();
      seedFileReferences.forEach(search.addToIndex);

      search.find('frank', (results) => {
           assert(results.length === 1);
           assert.equal(seedFileReferences[2].path, results[0].ref);
            done();
          });
           });
  });
});
```

190:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-16\lorikeet-nwjs\userInterface.js

```
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}
```

```
function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };

191:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cir
rus-nwjs\app.js
'use strict';



// Dependencies
//
const fs           = require('fs');
const gui          = require('nw.gui');
const designMenu   = require('./designMenu');
let currentFile;
let content;

const beetle = require('./beetle');


function openFile () {
     openFileDialog((filePath) => {
          fs.readFile(filePath, (err, data) => {
               setContent(data);
               hideSelectFileButton();
               showViewMode('design');
          });
     });
}



function saveFile () {
     fs.writeFile(currentFile, content, (err) => {
        if (err) {
            alert('There was an error');
        }
    });
}



function loadMenu () {

    const menuBar = new gui.Menu({type:'menubar'});

    // Create sub-menu
    const menuItems = new gui.Menu();

    menuItems.append(new gui.MenuItem({ label: 'Open', click: openFile }));
    menuItems.append(new gui.MenuItem({ label: 'Save', click: saveFile }));


    if (process.platform === 'darwin') {

        // Load Mac OS X application menu
        menuBar.createMacBuiltin('Cirrus');

        menuBar.insert(
            new gui.MenuItem({
                label: 'File',
                submenu: menuItems // menu elements from menuItems object
            }), 1
        );

    } else {

        // Load Windows/Linux application menu
        menuBar.append(
            new gui.MenuItem({
                label: 'File',
                submenu: menuItems // menu elements from menuItems object
            }), 1
        );

    }

    gui.Window.get().menu = menuBar;

}
```

```
function openFileDialog (cb) {
     const inputField = document.querySelector('#fileSelector');
     inputField.addEventListener('change', function () {
          const filePath = this.value;
          currentFile = filePath;
          cb(filePath);
     });
     inputField.click();
}



function bindSelectFileClick (cb) {
     const button = document.querySelector('#openFileView div');
     button.addEventListener('click', () => {
          openFileDialog(cb);
     });
}



function hideSelectFileButton () {
     const button = document.querySelector('#openFileView');
     button.classList.add('hidden');
     const appView = document.querySelector('#appView');
     appView.classList.remove('hidden');
}



function showViewMode (viewMode) {
     const areaDivs = document.querySelectorAll('.area');
     for (let i=0;i<areaDivs.length;i++) {
          let areaDiv = areaDivs[i];
          areaDiv.classList.add('hidden');
     }
     const selectedArea = document.querySelector(`#${viewMode}Area`);
     selectedArea.classList.remove('hidden');
}



function setContent (changedContent) {
     if (changedContent) { content = changedContent; }
     const designArea = document.querySelector('#designArea');
```

```
	designArea.innerHTML = content;
	const codeArea = document.querySelector('#codeArea');
	codeArea.value = content;
	const previewArea = document.querySelector('#previewArea');
	previewArea.innerHTML = content;
}

function initialize () {
	bindSelectFileClick((filePath) => {
		loadMenu();
		fs.readFile(filePath, (err, data) => {
			setContent(data);
			hideSelectFileButton();
			showViewMode('design');
		});
	});
	designMenu(window, gui);
}



window.onload = initialize;

192:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cir
rus-nwjs\beetle.js
check.line;

193:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cir
rus-nwjs\designMenu.js
'use strict';

let x;
let y;
let document;

function insertContent (content) {
	const range = document.caretRangeFromPoint(x, y);
	if (range) {
	  range.insertNode(content);
	}
}

function openImageFileDialog (cb) {
	const inputField = document.querySelector('#imageFileSelector');
	inputField.addEventListener('change', () => {
		const filePath = this.value;
		cb(filePath);
```

```
    });
    inputField.click();
}

function insertImage () {
    openImageFileDialog((filePath) => {
        if (filePath !== '') {
            const newImageNode = document.createElement('img');
            newImageNode.src = filePath;
            insertContent(newImageNode);
        }
    });
}

function parseYoutubeVideo (youtubeURL) {
    if (youtubeURL.indexOf('youtube.com/watch?v=') > -1) {
        return youtubeURL.split('watch?v=')[1];
    } else if (youtubeURL.match('https://youtu.be/') !== null) {
        return youtubeURL.split('https://youtu.be/')[1];
    } else if (youtubeURL.match('<iframe') !== null) {
        return youtubeURL.split('youtube.com/embed/')[1].split('"')[0];
    } else {
        alert('Unable to find a YouTube video id in the url');
        return false;
    }
}

function insertVideo () {
    const youtubeURL = prompt('Please insert a YouTube url');
    if (youtubeURL) {
        const videoId = parseYoutubeVideo(youtubeURL);
        if (videoId) {
            const newIframeNode = document.createElement('iframe');
            newIframeNode.width = 854;
            newIframeNode.height = 480;
            newIframeNode.src = 'https://www.youtube.com/embed/' + videoId;
            newIframeNode.frameborder = 0;
            newIframeNode.allowfullscreen = true;
            insertContent(newIframeNode);
        }
    }
}

function initialize (window, gui) {

    if (!document) document = window.document;
    const menu = new gui.Menu();

    menu.append(new gui.MenuItem({icon: 'picture.png', label: 'Insert image', click:
insertImage }));
    menu.append(new gui.MenuItem({icon: 'youtube.png', label: 'Insert video', click:
insertVideo }));

    document.querySelector('#designArea')
    .addEventListener('contextmenu', (event) => {
        event.preventDefault();
        x = event.x;
        y = event.y;
        menu.popup(event.x, event.y);
        return false;
    });

}

module.exports = initialize;

194:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cir
rus-nwjs\index.html
<!doctype html>
<html lang="en">
    <head>
        <title>Cirrus</title>
        <link href="app.css" rel="stylesheet" />
        <script src="app.js"></script>
    </head>
    <body>
        <input type="file" accept="image/*" id="imageFileSelector"
class="hidden"/>
        <input type="file" accept=".html,.htm" id="fileSelector" class="hidden"/>
        <div id="openFileView">
            <div>Select a HTML file</div>
        </div>
        <div id="appView" class="hidden">
            <div id="toolbar">
                <div class="tab" id="design"
onclick="showViewMode('design');">Design</div>
                <div class="tab" id="code"
onclick="showViewMode('code');">Code</div>
                <div class="tab" id="preview"
onclick="showViewMode('preview');">Preview</div>
            </div>
            <div class="area hidden" id="designArea" contenteditable
onblur="setContent(this.innerHTML);"></div>
```

```
            <textarea         class="area         hidden"         id="codeArea"
onblur="setContent(this.value);"></textarea>
            <div class="area hidden" id="previewArea"></div>
        </div>
    </body>
</html>
```

195:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cirrus-nwjs\package.json

```
{
  "name": "cirrus",
  "version": "1.0.0",
  "main": "index.html",
  "window": {
    "icon": "cirrus-logo.png",
    "toolbar": true
  },
  "devDependencies": {
    "nw": "^0.12.0"
  }
}
```

196:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\cirrus-nwjs\README.md

```
# Cirrus (NW.js)
A WYSIWYG HTML editor, built with NW.js

### Installation

    npm install -g nw
    cd cirrus
    nw

### About Cirrus

This is the source code for one of the apps featured in ["Cross Platform Desktop Applications"](http://manning.com/books/cross-platform-desktop-applications).
```

197:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-electron\app.js

```
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

198:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-electron\fileSystem.js
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

199:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-electron\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
      <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>

200:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-electron\main.js
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${app.getAppPath()}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});

201:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-electron\package.json
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "main.js",
  "author": "Paul Jensen <paul@anephenix.com>",
  "description": "A file explorer application",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  },
  "scripts": {
    "pack": "build",
    "dist": "build"
  },
  "devDependencies": {
    "devtron": "^1.4.0",
    "electron": "^1.4.14",
    "electron-builder": "^11.4.4"
  },
  "build": {}
}

202:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-electron\search.js
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

203:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-electron\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}
```

```
function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
    }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}
```

```
function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };
```

204:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-nwjs\app.js

```
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;
```

205:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-nwjs\fileSystem.js

```
'use strict';
```

```
const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}

function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

206:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
              <div id="current-folder"></div>
              <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

207:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  }
}
```

208:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor

```
ikeet-nwjs\search.js
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };

209:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-17\lor
ikeet-nwjs\userInterface.js
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');

function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
          search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
          } else {
          clone.querySelector('img')
          .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
          },
          false);
     }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
```

```
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };
```

210:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\hello-world-electron\index.html

```
<html>
  <head>
    <title>Hello World</title>
    <style>
      body {
        background-image: linear-gradient(45deg, #EAD790 0%, #EF8C53 100%);
        text-align: center;
      }

      button {
        background: rgba(0,0,0,0.40);
        box-shadow: 0px 0px 4px 0px rgba(0,0,0,0.50);
        border-radius: 8px;
        color: white;
        padding: 1em 2em;
        border: none;
        font-family: 'Roboto', sans-serif;
        font-weight: 300;
        font-size: 14pt;
        position: relative;
        top: 40%;
        cursor: pointer;
        outline: none;
      }

      button:hover {
        background: rgba(0,0,0,0.30);
      }
    </style>
    <link               href='https://fonts.googleapis.com/css?family=Roboto:300'
rel='stylesheet' type='text/css' />
    <script>
      function sayHello () {
        alert('Hello World');
      }
    </script>
  </head>
  <body>
    <button onclick="sayHello()">Say Hello</button>
  </body>
</html>
```

211:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\hello-world-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

212:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\hello-world-electron\package.json

```
{
  "name": "hello-world",
  "description": "A hello world Electron application",
  "version": "1.0.0",
  "author": "Paul Jensen <paul@anephenix.com>",
  "main": "main.js",
  "build": {
    "iconUrl": " https://github.com/paulbjensen/lorikeet/raw/master/icon.ico",
         "max": {
      "title": "Hello World",
      "icon": "icon.icns",
      "background": "background.png",
      "icon-size": 80,
      "contents": [
        {
          "x": 448,
          "y": 220,
          "type": "link",
          "path": "/Applications"
        },
        {
          "x": 192,
          "y": 220,
          "type": "file",
          "path": "dist/hello-world-darwin-x64/hello-world.app"
        }
      ]
    }
  },
  "scripts": {
    "pack": "build",
    "dist": "build"
  },
  "dependencies": {},
  "devDependencies": {
    "electron": "^1.4.15",
    "electron-builder": "^13.5.0"
  }
}

213:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lor
ikeet-nwjs\app.js
'use strict';

const fileSystem = require('./fileSystem');
const userInterface = require('./userInterface');
const search = require('./search');

function main() {
  userInterface.bindDocument(window);
  let folderPath = fileSystem.getUsersHomeFolder();
  userInterface.loadDirectory(folderPath)(window);
  userInterface.bindSearchField((event) => {
    const query = event.target.value;
    if (query === '') {
      userInterface.resetFilter();
    } else {
      search.find(query, userInterface.filterResults);
    }
  });
}

window.onload = main;

214:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lor
ikeet-nwjs\fileSystem.js
'use strict';

const async = require('async');
const fs = require('fs');
const osenv = require('osenv');
const path = require('path');

let shell;

if (process.versions.electron) {
  shell = require('electron').shell;
} else {
  shell = window.require('nw.gui').Shell;
}

function getUsersHomeFolder() {
  return osenv.home();
}

function getFilesInFolder(folderPath, cb) {
  fs.readdir(folderPath, cb);
}

function inspectAndDescribeFile(filePath, cb) {
  let result = { file: path.basename(filePath), path: filePath, type: '' };
  fs.stat(filePath, (err, stat) => {
    if (err) {
      cb(err);
    }       else {
      if (stat.isFile()) {
        result.type = 'file';
      }
      if (stat.isDirectory()) {
        result.type = 'directory';
      }
      cb(err, result);
    }
  });
}
```

```
function inspectAndDescribeFiles(folderPath, files, cb) {
  async.map(files, (file, asyncCb) => {
    let resolvedFilePath = path.resolve(folderPath, file);
    inspectAndDescribeFile(resolvedFilePath, asyncCb);
  }, cb);
}

function openFile(filePath) {
  shell.openItem(filePath);
}

module.exports = {
  getUsersHomeFolder,
  getFilesInFolder,
  inspectAndDescribeFiles,
     openFile
};
```

215:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lorikeet-nwjs\index.html

```
<html>
  <head>
    <title>Lorikeet</title>
    <link rel="stylesheet" href="app.css" />
    <script src="app.js"></script>
  </head>
  <body>
    <template id="item-template">
      <div class="item">
        <img class="icon" />
        <div class="filename"></div>
      </div>
    </template>
    <div id="toolbar">
            <div id="current-folder"></div>
            <input type="search" id="search" results="5" placeholder="Search" />
    </div>
    <div id="main-area"></div>
  </body>
</html>
```

216:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lorikeet-nwjs\package.json

```
{
  "name": "lorikeet",
  "version": "1.0.0",
  "main": "index.html",
  "dependencies": {
    "async": "^2.1.4",
    "lunr": "^0.7.2",
    "osenv": "^0.1.4"
  }
}
```

217:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lorikeet-nwjs\search.js

```
'use strict';

const lunr = require('lunr');
let index;

function resetIndex() {
  index = lunr(function () {
    this.field('file');
    this.field('type');
    this.ref('path');
  });
}

function addToIndex(file) {
  index.add(file);
}

function find(query, cb) {
  if (!index) {
    resetIndex();
  }

  const results = index.search(query);
  cb(results);
}

module.exports = { addToIndex, find, resetIndex };
```

218:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chapter-18\lorikeet-nwjs\userInterface.js

```
'use strict';

let document;
const fileSystem = require('./fileSystem');
const search = require('./search');
const path = require('path');
```

```
function displayFolderPath(folderPath) {
  document.getElementById('current-folder')
    .innerHTML = convertFolderPathIntoLinks(folderPath);
  bindCurrentFolderPath();
}

function clearView() {
  const mainArea = document.getElementById('main-area');
  let firstChild = mainArea.firstChild;
  while (firstChild) {
    mainArea.removeChild(firstChild);
    firstChild = mainArea.firstChild;
  }
}

function loadDirectory(folderPath) {
  return function (window) {
    if (!document) document = window.document;
         search.resetIndex();
    displayFolderPath(folderPath);
    fileSystem.getFilesInFolder(folderPath, (err, files) => {
      clearView();
      if (err) {
        return alert('Sorry, we could not load your folder');
      }
      fileSystem.inspectAndDescribeFiles(folderPath, files, displayFiles);
    });
  };
}

function displayFile(file) {
  const mainArea = document.getElementById('main-area');
  const template = document.querySelector('#item-template');
  let clone = document.importNode(template.content, true);
  search.addToIndex(file);
  clone.querySelector('img').src = `images/${file.type}.svg`;
  clone.querySelector('img').setAttribute('data-filePath', file.path);
  if (file.type === 'directory') {
    clone.querySelector('img')
      .addEventListener('dblclick', () => {
        loadDirectory(file.path)();
      }, false);
         } else {
         clone.querySelector('img')
         .addEventListener('dblclick', () => {
        fileSystem.openFile(file.path);
         },
          false);
      }
  clone.querySelector('.filename').innerText = file.file;
  mainArea.appendChild(clone);
}


function displayFiles(err, files) {
  if (err) {
    return alert('Sorry, we could not display your files');
  }
  files.forEach(displayFile);
}

function bindDocument (window) {
  if (!document) {
    document = window.document;
  }
}

function bindSearchField(cb) {
  document.getElementById('search').addEventListener('keyup', cb, false);
}

function filterResults(results) {
  const validFilePaths = results.map((result) => { return result.ref; });
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    let item = items[i];
    let filePath = item.getElementsByTagName('img')[0]
      .getAttribute('data-filepath');
    if (validFilePaths.indexOf(filePath) !== -1) {
      item.style = null;
    } else {
      item.style = 'display:none;';
    }
  }
}

function resetFilter() {
  const items = document.getElementsByClassName('item');
  for (var i = 0; i < items.length; i++) {
    items[i].style = null;
  }
}

function convertFolderPathIntoLinks (folderPath) {
  const folders = folderPath.split(path.sep);
  const contents    = [];
  let pathAtFolder = '';
  folders.forEach((folder) => {
     pathAtFolder += folder + path.sep;
     contents.push(`<span                                             class="path"
data-path="${pathAtFolder.slice(0,-1)}">${folder}</span>`);
  });
  return contents.join(path.sep).toString();
}

function bindCurrentFolderPath() {
  const load = (event) => {
    const folderPath = event.target.getAttribute('data-path');
    loadDirectory(folderPath)();
  };

  const paths = document.getElementsByClassName('path');
  for (var i = 0; i < paths.length; i++) {
    paths[i].addEventListener('click', load, false);
  }
}

module.exports = { bindDocument, displayFiles, loadDirectory, bindSearchField,
filterResults, resetFilter };

219:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hell
o-world-electron\index.html
<html>
  <head>
    <title>Hello World</title>
    <style>
      body {
        background-image: linear-gradient(45deg, #EAD790 0%, #EF8C53 100%);
        text-align: center;
      }

      button {
        background: rgba(0,0,0,0.40);
        box-shadow: 0px 0px 4px 0px rgba(0,0,0,0.50);
        border-radius: 8px;
        color: white;
        padding: 1em 2em;
        border: none;
        font-family: 'Roboto', sans-serif;
        font-weight: 300;
        font-size: 14pt;
        position: relative;
        top: 40%;
        cursor: pointer;
        outline: none;
      }


      #foo{
        width: 1200px;
        height: 1000px;
      }

      button:hover {
        background: rgba(0,0,0,0.30);
      }
    </style>
    <link                href='https://fonts.googleapis.com/css?family=Roboto:300'
rel='stylesheet' type='text/css' />
    <script>
      function sayHello () {
        alert('Hello World');
      }
    </script>
  </head>
  <body>
    <!-- <button onclick="sayHello()">Say Hello</button> -->

    <webview          id="foo"          src="http://localhost:8080/boot/doc/index"
style="display:inline-block; width:1200px; height:1000px"></webview>


  </body>
</html>
```

220:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hello-world-electron\main.js

```
'use strict';

const electron = require('electron');
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;

let mainWindow = null;

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

```
app.on('ready', () => {
  mainWindow = new BrowserWindow();
  //mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.loadURL('http://localhost:8080/boot/doc/index');
  mainWindow.on('closed', () => { mainWindow = null; });
});
```

221:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hello-world-electron\package-lock.json

```
{
    "name": "hello-world",
    "version": "1.0.0",
    "lockfileVersion": 1
}
```

222:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hello-world-electron\package.json

```
{
    "name": "hello-world",
    "version": "1.0.0",
    "main": "main.js"
}
```

223:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hello-world-nwjs\index.html

```
<html>
    <head>
        <title>Hello World</title>
        <style>
          body {
                background-image: linear-gradient(45deg, #EAD790 0%, #EF8C53 100%);
                text-align: center;
            }

            button {
                background: rgba(0,0,0,0.40);
                box-shadow: 0xp 0px 4px 0px rgba(0,0,0,0.50);
                border-radius: 8px;
                color: white;
                padding: 1em 2em;
                border: none;
                font-family: 'Roboto', sans-serif;
                font-weight: 100;
                font-size: 14pt;
                position: relative;
```

```
                    top: 40%;
                    cursor: pointer;
                    outline: none;
               }

               button:hover {
                    background: rgba(0,0,0,0.30);
               }
          </style>
          <link           href='https://fonts.googleapis.com/css?family=Roboto:300'
rel='stylesheet' type='text/css'>
          <script>
            function sayHello () {
                    alert('Hello World');
               }
          </script>
     </head>
     <body>
          <button onclick="sayHello()">Say Hello</button>
     </body>
</html>
```

224:F:\git\nodejs\electron-book\cross-platform-desktop-applications\chaptermy\hello-world-nwjs\package.json

```
{
  "name": "hello-world-nwjs",
  "main": "index.html",
  "version": "1.0.0"
}
```