Silex は PHP 製のマイクロフレームワークで、EC-CUBE 3 のベースとしても利用されています。インストールには Composer が必要となりますのであらかじめインストールしておいて下さい。詳細は割愛しますが Linux, Unix, macOS であれば下記のようにしてインストールします。Windows の場合 exe 形式のインストーラーが公式に用意されています。

```
curl -s http://getcomposer.org/installer | php
sudo mv composer.phar /usr/local/bin/composer
```

Silex 2.0のインストール

Silex のインストール方法は 2 種類あり、公式サイトからアーカイブファイルをダウンロードする方法と Composer 経由でインストールする方法があります。アーカイブファイルには必要最低限の slim 版と、全て入った fat 版がありますが、必要に応じてあとから追加していくこともできるので slim 版を使うのが良いと思います。今回は「silex-app」というフォルダ名で作業を進めていきますのでアーカイブ版を使う場合は展開してフォルダ名を変更して下さい。プロジェクトフォルダ名は自由につけても構いません。

Composer 経由でインストールする場合次のようにします。プロジェクト用のフォルダ「silex-app」を作り、そのフォルダ内にインストールします。（フォルダ名は一例ですので自由な名前をつけて読み替えて下さい。）

```
mkdir silex-app
cd silex-app
composer require silex/silex "~2.0"
```

これで Silex 本体はインストールされました。他にもテンプレートエンジンの Twig などが必要となりますので Composer で追加します。

```
composer require twig/twig
composer require symfony/twig-bridge
```

クラスファイルの自動読み込み(PSR-4)に対応するため、composer.json ファイルに手を加えます。
環境によってバージョンの違いはあると思いますが最終的に次のようになりました。

```
{
    "require": {
        "silex/silex": "~2.0",
        "twig/twig": "^1.24",
        "symfony/twig-bridge": "^3.1"
    },
    "autoload": {
        "psr-4": {
            "App\\": "app/"
        }
    }
}
```

書き足したら「composer dump-autoload」コマンドで更新しておきます。

コントローラー、ビューの作成

次に現在のプロジェクトフォルダに「app」と「web」の２つのフォルダを作ります。アーカイブ版をダウンロードした場合すでに「web」フォルダがあるかもしれません。「web」フォルダは Web 公開フォルダとなりますのでドメインの割り当てる際はこのフォルダをルートにして下さい。画像ファイルや CSS、JS ファイルもこのフォルダ以下に置きます。

フォルダを作成したら「web」フォルダ内に移動し、「index.php」ファイルと、「.htaccess」ファイル(Apacheの場合)を作成して下さい。
ファイルの内容は次のとおりです。

web\index.php

```php
<?php
require_once __DIR__.'/../vendor/autoload.php';

$app = new Silex\Application();
$app['debug'] = true;
$app->get('/', function(){ return 'Hello, World!'; });

$app->run();
```

web\.htaccess

```
<IfModule mod_rewrite.c>
    Options -MultiViews

    RewriteEngine On
    #RewriteBase /silex-app/web
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteRule ^ index.php [QSA,L]
</IfModule>
```

環境によっては RewriteBase を書き換える必要があります。# を外して適宜編集して下さい。

用意ができたらサイトにWebブラウザでアクセスします。ドメインを利用しない場合
「http://localhost/silex-app/web」のようなアドレスになっていると思います。正しく動作している場合ブラウザに「Hello, World!」の文字が表示されます。

「$app->get()」となっている部分がルーティング処理です。「/」にアクセスしたときの動作を定義しています。
単純なプログラムでない限りは例のようにせず、ページコントローラーを経由してビューを表示することになります。
まずはコントローラーのためのフォルダを作成します。「app\Controllers」となるように作成して下さい。
その中にファイル「HomeController.php」を作成して下さい。

app\Controllers\HomeController.php

```php
<?php
```

```php
namespace App\Controllers;

use Silex\Application;
use Symfony\Component\HttpFoundation\Request;

class HomeController
{
    public function index(Application $app, Request $request){
        $data = ['foo' => 'World'];
        return $app['twig']->render('index.twig', $data);
    }
}
```

render() によって表示するビューファイルを指定し、第２引数でビュー内で使える変数を連想配列の形で渡します。必要なければ空でも構いません。
上記の例では変数 foo に 値「World」を格納してビューに渡しています。

次にビューとなる Twig ファイルを用意します。プロジェクトフォルダに「views」フォルダを作成し、その中に index.twig ファイルを作成します。

views\index.twig

```twig
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Silex</title>
    </head>
    <body>
        <h1>Hello, {{foo}}!</h1>
    </body>
</html>
```

基本的には HTML ファイルと変わりません。{{ 〜 }} を使うことで変数の内容などを表示します。この内容は自動的に安全な文字列に変換（エスケープ）されます。

コントローラーとビューが用意できたら Twig を使うための準備とルーティングを行います。再度 web\index.php を編集して下さい。twig.path には twig ファイルが格納されているフォルダのパスを指定します。

```php
$app->register(new Silex\Provider\TwigServiceProvider(), [
    'twig.path' => __DIR__ . '/../views',
]);

$app->get('/', 'App\Controllers\HomeController::index')->bind('index');
```

get() の箇所を上のように書き換えます。これにより「/」にアクセスした際は HomeController のメソッド index() が実行されます。bind() によって一意な名前をつけておくと URL を生成する際に役に立ちます。ビューで下記のように名前を指定するとそのページへの URL が生成されます。

```twig
{{ url("index") }}
```

user/1234、post/1234 のようにユーザーIDや記事IDを URL からパラメータとして受け取る場合のルーティングは次のようにします。

```
$app->get('/post/{id}', 'App\Controllers\PostController::show')->assert('id', '\d+')->bind('po
```

id は波括弧で括ります。assert() の正規表現によって id が数値のみである場合という条件にしています。
コントローラーでは次のようにして id の値を受け取れます。bind() ではアンダーバーで区切って「post_show」と名付けました。

app\Controllers\PostController.php

```
1  public function show(Application $app, Request $request, $id){
2      //
3  }
```

引数 $id の中に ID の値が格納されます。この場合に Twig ビューで URL を生成するには次のようにします。

```
{{ url('post_show', {'id': 1234}) }}
```

コントローラーで「/?category=1」のような GET や POST パラメーターを受け取るには Request を使います。

```
$request->get('category');
```

以上が基本的な流れとなります。ここで紹介したクラス名やファイルの設置場所はあくまで例であり、どんな名前をつけるか、どこに置くかについてはアプリケーションに応じて変更して下さい。

Twigの詳細な書き方についてはこちらをお読み下さい。