Using SD Everywhere to Bring SD Models to Mac, Windows, and the Web

By Travis Franck and Todd Fincannon

Abstract

A good user interface makes System Dynamics models come alive for policymakers and stakeholders. This workshop will show how Climate Interactive translates large Vensim models into C and JavaScript code which can be embedded in custom applications. The open source SDEverywhere toolkit developed by Climate Interactive and Todd Fincannon gives you full control over user interface development. We will show how to get SDEverywhere, translate your model, and run it everywhere: on Mac, Windows, and the web. SDEverywhere is one of several new community tools (e.g., SDXchange project, PySD) that are providing open source solutions to make SD models more useful.

Visit sdeverywhere.org for the latest updates, tutorials, release information, and news.

Required software

SDEverywhere currently runs on macOS. You will need a Mac and a text editor such as Atom.

We will be working from the command line. You can use the Terminal app that is built into macOS, or try the iTerm2 terminal emulator.

The instructions below will guide you through installing the other tools required by SDEverywhere.

- Xcode
- Node.js
- Emscripten SDK

Set up SDEverywhere

Follow this procedure to set up SDEverywhere on your macOS computer.

1. If you have installed Apple's Xcode development kit on your Mac, skip this step. Otherwise, open the App Store app on your Mac, search for `Xcode`, download it, and install it.

2. If you have installed Node.js version 6.11.1 LTS or later, skip this step. Otherwise, use n-install to install the latest version of Node.

3. Install the SDEverywhere Node package from the command line.

```
npm install sdeverywhere -g
```

4. Download SDEverywhere source code from the SDEverywhere GitHub repository. Unzip it to a convenient location on your computer.

5. Install a mini web server with the Node Package Manager. We need this to run the web demo.

```
npm install http-server -g
```

6. Install the Portable Emscripten SDK for OS X.

7. Edit the `emsdk_set_env.sh` file that was just created to remove the `clang` and `node` directories from the PATH. (They are second and third directories in the list.) The `...` below is a placeholder for the folder where you installed Emscripten.

   ```
   .../emsdk-portable/clang/e1.37.16_64bit
   .../emsdk-portable/node/4.1.1_64bit/bin
   ```

8. Close your terminal window. Reopen it, go back to the `emsdk-portable` directory, and enable the Emscripten environment. You can put this command in your `~/.bash_profile` if you want to permanently enable Emscripten.

   ```
   source emsdk_set_env.sh
   ```

Generate and test model code in C

1. We will be using the `lotka` sample model in the SDEverywhere `models` directory. From the command line, go the SDEverywhere directory, and then the `lotka` directory.

   ```
   cd models/lotka
   ```

2. Generate C code from the model. It will be written to a new `build` folder.

   ```
   sde generate --genc lotka
   ```

3. Compile the model code with the C compiler. This makes a `lotka` app in the `build` folder.

   ```
   sde compile lotka
   ```

4. Execute the `lotka` app. This writes run data to a tab-delimited text file in a new `output` folder.

   ```
   sde exec lotka
   ```

5. Convert the text file to Vensim DAT format.

   ```
   sde log --dat output/lotka.txt
   ```

6. Verify the C model by comparing its output to a DAT file exported from Vensim. Any discrepancies will be printed.

```
sde compare lotka.dat output/lotka.dat
```

7. Delete the build and output directories.

```
sde clean lotka
```

8. Generate, compile, execute, log, and compare with one command.

```
sde test lotka
```

Generate WebAssembly code and a web user interface

1. Clean out the code from the previous build.

```
sde clean lotka
```

2. Generate WebAssembly code that the web interface will run.

```
sde generate --genwebc --spec lotka_web_spec.json lotka
```

3. Compile the WebAssembly code. If you see a popup about installing Java, click OK to close it.

```
sde compile --wasm lotka
```

4. Generate an HTML user interface using the web specification JSON file.

```
sde generate --genhtml --spec lotka_web_spec.json lotka
```

5. Go to the `html` directory and run the mini web server.

```
cd html
http-server
```

6. Open the web page in your browser by entering `127.0.0.1:8080` in the address bar.

7. Back in Terminal, press Ctrl+C to stop the web server.

Contact

Todd Fincannon
todd@toddfincannon.com

Travis Franck
tfranck@climateinteractive.org

Visit sdeverywhere.org for the latest updates.