Community Experience Distilled

SDL Game Development

Discover how to leverage the power of SDL 2.0 to create awesome games in C++

Shaun Mitchell

[PACKT] PUBLISHING

SDL Game Development

Discover how to leverage the power of SDL 2.0 to create awesome games in C++

Shaun Ross Mitchell

[PACKT]
PUBLISHING
BIRMINGHAM - MUMBAI

SDL Game Development

First published: June 2013

Production Reference: 1170613

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84969-682-1

www.packtpub.com

Cover Image by Shaun Mitchell (shaunmitchell84@googlemail.com)

Credits

Author
Shaun Ross Mitchell

Reviewers
Luka Horvat
Mårten Möller

Acquisition Editor
Edward Gordon

Lead Technical Editor
Savio Jose
Chalini Snega Victor

Technical Editors
Jeeten Handu
Kaustubh S. Mayekar
Anita Nayak

Project Coordinator
Hardik Patel

Proofreader
Bernadette Watkins

Indexer
Rekha Nair

Graphics
Ronak Dhruv

Production Coordinator
Prachali Bhiwandkar

Cover Work
Prachali Bhiwandkar

About the Author

Shaun Mitchell is a developer at a high profile online gaming company. He holds a BSc in Game Programming and Development from Qantm College / SAE Institute London. Shaun is also a moderator and active member of the <dream.in.code> programming community.

I would like to thank Jason Colman, my university lecturer, whose knowledge and insight into C++ and game programming has been the foundation of my skillset.

I would also like to thank the <dream.in.code> community for the interesting discussions and topics to hone my skills with.

Thank you to John Bayly for the background image on the front cover.

Many thanks to my family for their continued support and importantly, a huge thank you to my girlfriend, Emma, who tirelessly proofread my chapters while also keeping me running on a generous amount of caffeine.

About the Reviewers

Luka Horvat is an enthusiastic software and game developer who got fascinated by computer science in his early years. He chose to study his passion while working on many different projects and technologies. Throughout the years he gained a lot of knowledge and experience, and he wanted to share that with others. He is proficient in many different programming languages, with C++ as his main one; and is passionate about game development. So he started teaching it and currently manages different courses for in this area. He continues to pursue his career in computer science by working on a wide variety of projects and sharing them with others.

I would like to thank my friends and family who helped me produce this book.

Mårten Möller is an independent game developer who has previously worked at Imperial Game Studios.

I would like to thank my family and friends. All of you are amazing.

---

www.PacktPub.com

Support files, eBooks, discount offers and more

You might want to visit `www.PacktPub.com` for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`http://PacktLib.PacktPub.com`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

In memory of my Mum.
You always believed in me.
I miss you everyday.

Table of Contents

Preface

Creating games in C++ is a complicated process requiring a lot of time and dedication to achieve results. A good foundation of reusable classes can speed up development time and allow focus to be on creating a great game rather than struggling with low-level code. This book aims to show an approach to creating a reusable framework that could be used for any game, whether 2D or 3D.

What this book covers

Chapter 1, *Getting started with SDL*, covers setting up SDL in Visual C++ 2010 express and then moves onto the basics of SDL including creating a window and listening for quit events.

Chapter 2, *Drawing in SDL*, covers the development of some core drawing classes to help simplify SDL rendering. The `SDL_image` extension is also introduced to allow the loading of a variety of different image file types.

Chapter 3, *Working with Game Objects*, gives a basic introduction to inheritance and polymorphism along with the development of a reusable `GameObject` class that will be used throughout the rest of the book.

Chapter 4, *Exploring Movement and Input Handling*, gives a detailed look at handling events in SDL. Joystick, keyboard, and mouse input are all covered with the development of reusable classes.

Chapter 5, *Handling Game States*, covers the design and implementation of a finite state machine to manage game states. Implementing and moving between different states is covered in detail.

Chapter 6, *Data-driven Design*, covers the use of TinyXML to load states. A class to parse states is developed along with examples for different states.

Chapter 7, *Creating and Displaying Tile Maps*, brings together everything from the previous chapters to allow the creation of levels using the Tiled map editor. A level parsing class is created to load maps from an XML file.

Chapter 8, *Creating Alien Attack*, covers the creation of a 2D side scrolling shooter, utilizing everything learned in the previous chapters.

Chapter 9, *Creating Conan the Caveman*, covers the creation of a second game, altering the code from Alien Attack, showing that the framework is flexible enough to be used for any 2D game genre.

What you need for this book

To use this book you will need the following software:

- Visual C++ 2010 Express
- Tiled map editor
- TinyXML
- zlib library

Who this book is for

This book is aimed at beginner/intermediate C++ programmers who want to take their existing skills and apply them to creating games in C++. This is not a beginner's book and you are expected to know the basics of C++, including inheritance, polymorphism, and class design.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "We can include other contexts through the use of the `include` directive."

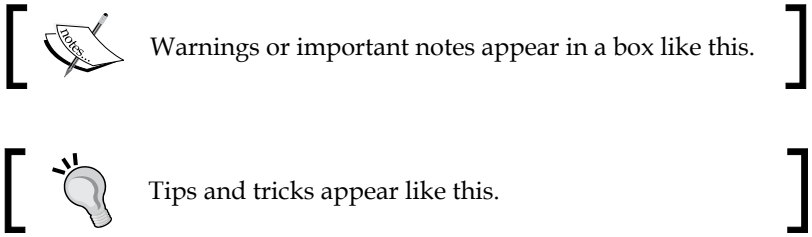
A block of code is set as follows:

```
void Player::update()
{
  m_currentFrame = int(((SDL_GetTicks() / 100) % 6));
```

```
    m_acceleration.setX(1);

    SDLGameObject::update();
  }
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Right-click on the project and choose **Build**.".



Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **erratasubmissionform** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Getting Started with SDL

Simple DirectMedia Layer (**SDL**) is a cross-platform multimedia library created by Sam Oscar Latinga. It provides low-level access to input (via mouse, keyboard, and gamepads/joysticks), 3D hardware, and the 2D video frame buffer. SDL is written in the C programming language, yet has native support for C++. The library also has bindings for several other languages such as Pascal, Objective-C, Python, Ruby, and Java; a full list of supported languages is available at `http://www.libsdl.org/languages.php`.

SDL has been used in many commercial games including World of Goo, Neverwinter Nights, and Second Life. It is also used in emulators such as ZSNES, Mupen64, and VisualBoyAdvance. Some popular games ported to Linux platforms such as Quake 4, Soldier of Fortune, and Civilization: Call to Power utilize SDL in some form.

SDL is not just used for games. It is useful for all manner of applications. If your software needs access to graphics and input, chances are that SDL will be a great help. The SDL official website has a list of applications that have been created using the library (`http://www.libsdl.org/applications.php`).

In this chapter we will cover the following:

- Getting the latest SDL build from the Mercurial repository
- Building and setting up SDL in Visual C++ 2010 Express
- Creating a window with SDL
- Implementing a basic game class

Why use SDL?

Each platform has its own way of creating and displaying windows and graphics, handling user input, and accessing any low-level hardware; each one with its own intricacies and syntax. SDL provides a uniform way of accessing these platform-specific features. This uniformity leads to more time spent tweaking your game rather than worrying about how a specific platform allows you to render or get user input, and so on. Game programming can be quite difficult, and having a library such as SDL can get your game up and running relatively quickly.

The ability to write a game on Windows and then go on to compile it on OSX or Linux with little to no changes in the code is extremely powerful and perfect for developers who want to target as many platforms as possible; SDL makes this kind of cross-platform development a breeze. While SDL is extremely effective for cross-platform development, it is also an excellent choice for creating a game with just one platform in mind, due to its ease of use and abundance of features.

SDL has a large user base and is being actively updated and maintained. There is also a responsive community along with a helpful mailing list. Documentation for SDL 2.0 is up-to-date and constantly maintained. Visiting the SDL website, `libsdl.org`, offers up lots of articles and information with links to the documentation, mailing list, and forums.

Overall, SDL offers a great place to start with game development, allowing you to focus on the game itself and ignore which platform you are developing for, until it is completely necessary. Now, with SDL 2.0 and the new features it brings to the table, SDL has become an even more capable library for game development using C++.

> The best way to find out what you can do with SDL and its various functions is to use the documentation found at `http://wiki.libsdl.org/moin.cgi/CategoryAPI`. There you can see a list of all of SDL 2.0's functions along with various code examples.

What is new in SDL 2.0?

The latest version of SDL and SDL 2.0, which we will be covering in this book, is still in development. It adds many new features to the existing SDL 1.2 framework. The SDL 2.0 Roadmap (`wiki.libsdl.org/moin.cgi/Roadmap`) lists these features as:

- A 3D accelerated, texture-based rendering API
- Hardware-accelerated 2D graphics
- Support for render targets

- Multiple window support
- API support for clipboard access
- Multiple input device support
- Support for 7.1 audio
- Multiple audio device support
- Force-feedback API for joysticks
- Horizontal mouse wheel support
- Multitouch input API support
- Audio capture support
- Improvements to multithreading

While not all of these will be used in our game-programming adventures, some of them are invaluable and make SDL an even better framework to use to develop games. We will be taking advantage of the new hardware-accelerated 2D graphics to make sure our games have excellent performance.

Migrating SDL 1.2 extensions

SDL has separate extensions that can be used to add new capabilities to the library. The reason these extensions are not included in the first place is to keep SDL as lightweight as possible, with the extensions serving to add functionality only when necessary. The next table shows some useful extensions along with their purpose. These extensions have been updated from their SDL1.2/3 Versions to support SDL 2.0, and this book will cover cloning and building them from their respective repositories as and when they are needed.

Name	Description
`SDL_image`	This is an image file loading library with support for BMP, GIF, PNG, TGA, PCX, and among others.
`SDL_net`	This is a cross-platform networking library.
`SDL_mixer`	This is an audio mixer library. It has support for MP3, MIDI, and OGG.
`SDL_ttf`	This is a library supporting the use of `TrueType` fonts in SDL applications.
`SDL_rtf`	This is a library to support the rendering of the **Rich Text Format** (**RTF**).

Setting up SDL in Visual C++ Express 2010

This book will cover setting up SDL 2.0 in Microsoft's Visual C++ Express 2010 IDE. This IDE was chosen as it is available for free online, and is a widely used development environment within the games industry. The application is available at `https://www.microsoft.com/visualstudio/en-gb/express`. Once the IDE has been installed we can go ahead and download SDL 2.0. If you are not using Windows to develop games, then these instructions can be altered to suit your IDE of choice using its specific steps to link libraries and include files.

SDL 2.0 is still in development so there are no official releases as yet. The library can be retrieved in two different ways:

- One is to download the under-construction snapshot; you can then link against this to build your games (the quickest option)
- The second option is to clone the latest source using mercurial-distributed source control and build it from scratch (a good option to keep up with the latest developments of the library)

Both of these options are available at `http://www.libsdl.org/hg.php`.

Building SDL 2.0 on Windows also requires the latest DirectX SDK, which is available at `http://www.microsoft.com/en-gb/download/details.aspx?id=6812`, so make sure this is installed first.

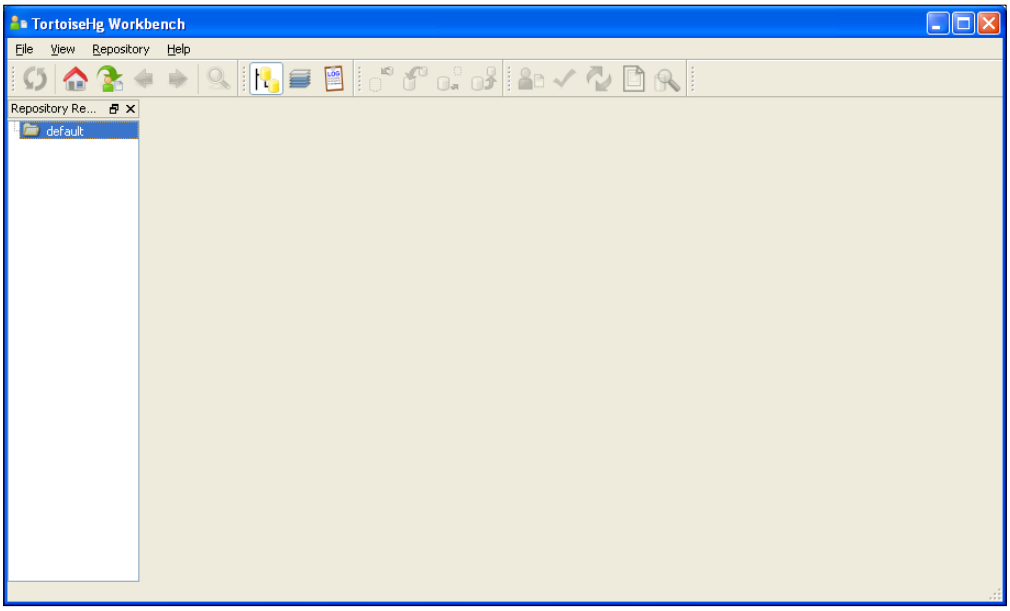
Using Mercurial to get SDL 2.0 on Windows

Getting SDL 2.0 directly from the constantly updated repository is the best way of making sure you have the latest build of SDL 2.0 and that you are taking advantage of any current bug fixes. To download and build the latest version of SDL 2.0 on Windows, we must first install a mercurial source control client so that we can mirror the latest source code and build from it. There are various command-line tools and GUIs available for use with mercurial. We will use TortoiseHg, a free and user-friendly mercurial application; it is available at `tortoisehg.bitbucket.org`. Once the application is installed, we can go ahead and grab the latest build.

Cloning and building the latest SDL 2.0 repository

Cloning and building the latest version of SDL directly from the repository is relatively straightforward when following these steps:

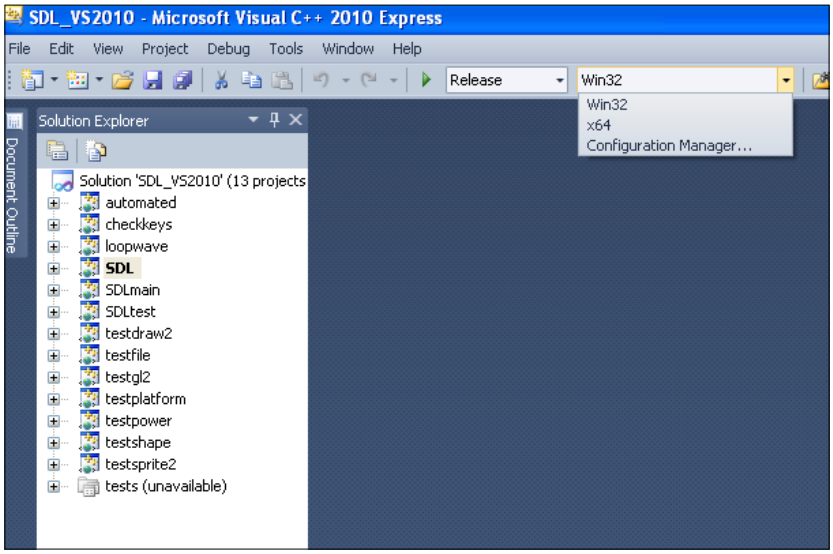
1. Open up the **TortoiseHg Workbench** window.



2. Pressing *Ctrl* + *Shift* + *N* will open the clone dialog box.
3. Input the source of the repository; in this case it is listed on the SDL 2.0 website as `http://hg.libsdl.org/SDL`.
4. Input or browse to choose a destination for the cloned repository—this book will assume that `C:\SDL2` is set as the location.
5. Click on **Clone** and allow the repository to copy to the chosen destination.

6. Within the `C:\SDL2` directory there will be a `VisualC` folder; inside the folder there is a Visual C++ 2010 solution, which we have to open with Visual C++ Express 2010.
7. Visual C++ Express will throw up a few errors about solution folders not being supported in the express version, but they can be safely ignored without affecting our ability to build the library.

8. Change the current build configuration to release and also choose 32 or 64 bit depending on your operating system.



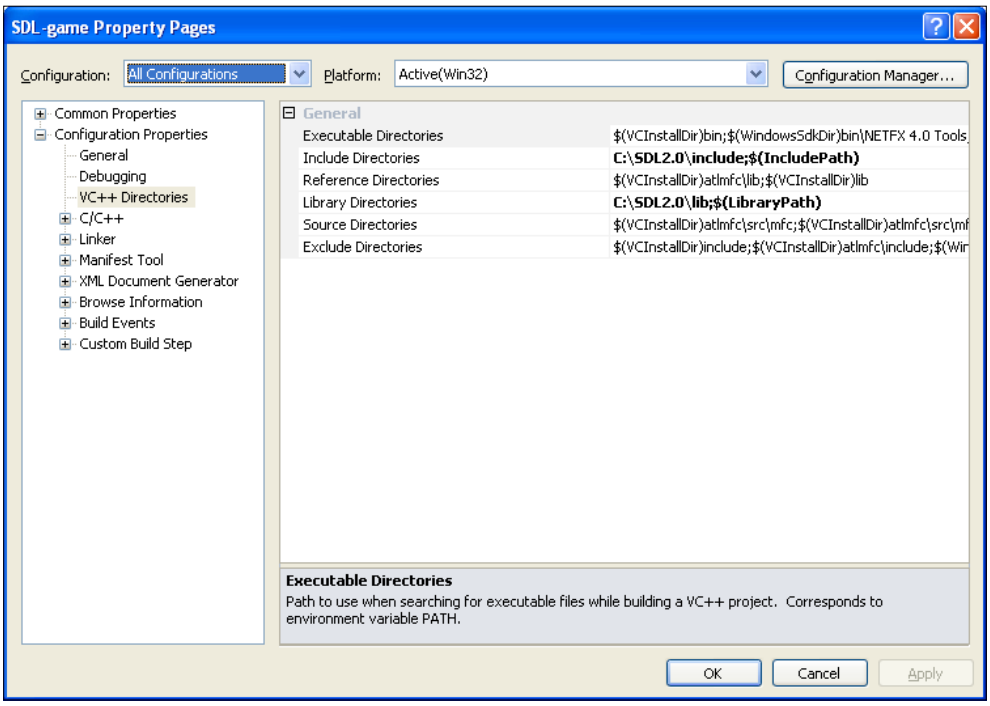
9. Right-click on the project named **SDL** listed in the **Solution Explorer** list and choose **Build**.
10. We now have a build of the SDL 2.0 library to use. It will be located at `C:\SDL2\VisualC\SDL\Win32(or x64)\Release\SDL.lib`.
11. We also need to build the SDL main library file, so choose it within the **Solution Explorer** list and build it. This file will build to `C:\SDL2\VisualC\SDLmain\Win32(or x64)\Release\SDLmain.lib`.
12. Create a folder named `lib` in `C:\SDL2` and copy `SDL.lib` and `SDLmain.lib` into this newly created folder.

I have the library; now what?

Now a Visual C++ 2010 project can be created and linked with the SDL library. Here are the steps involved:

1. Create a new empty project in Visual C++ express and give it a name, such as `SDL-game`.
2. Once created, right-click on the project in the **Solution Explorer** list and choose **Properties**.

3. Change the configuration drop-down list to **All Configurations**.
4. Under **VC++ Directories**, click on **Include Directories**. A small arrow will allow a drop-down menu; click on **<Edit…>**.



5. Double-click inside the box to create a new location. You can type or browse to `C:\SDL2.0\include` and click on **OK**.
6. Next, do the same thing under library directories, this time passing in your created `lib` folder (`C:\SDL2\lib`).