Introduction to Apple’s Developer Tools

Kevin Cathey
Introduction to Apple’s Developer Tools

acm.uiuc.edu/macwarriors/devphone
Introduction to Apple’s Developer Tools

The Xcode tools in the software development cycle

acm.uiuc.edu/macwarriors/devphone
Introduction to Apple’s Developer Tools

The Xcode tools in the software development cycle

Using Xcode to write code

acm.uiuc.edu/macwarriors/devphone
Introduction to Apple’s Developer Tools

The Xcode tools in the software development cycle

Using Xcode to write code

Using Xcode to debug and test code

acm.uiuc.edu/macwarriors/devphone
Introduction to Apple’s Developer Tools

The Xcode tools in the software development cycle
Using Xcode to write code
Using Xcode to debug and test code
Introduction to Interface Builder

acm.uiuc.edu/macwarriors/devphone
Introduction to Apple’s Developer Tools

- The Xcode tools in the software development cycle
- Using Xcode to write code
- Using Xcode to debug and test code
- Introduction to Interface Builder
- Introduction to profiling code with Instruments

acm.uiuc.edu/macwarriors/devphone
Software Development Cycle
And how the tools integrate with it
Software Development Cycle
And how the tools integrate with it

Start with a project
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface

Code

Implement

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code
Build
Implement

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface

- Code
- Build
- Debug

Implement
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface

Code ➔ Build ➔ Debug ➔ Implement

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code Build Debug Write tests Implement
Software Development Cycle
And how the tools integrate with it

1. Start with a project
2. Make core classes
3. Design an interface
4. Code
5. Build
6. Debug
7. Write tests
8. Implement
9. Profile
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code 
Build 
Debug 
Write tests 
Implement 

Profile

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface
Code Build Debug Write tests Implement
Profile

Start with a project
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code → Build → Debug → Write tests → Implement
Profile
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Write tests
- Build
- Debug
- Code
- Profile
- Implement

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Code
- Build
- Debug
- Write tests
- Implement
- Profile

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface

Code → Build → Debug → Write tests → Implement

Profile
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Code
- Build
- Debug
- Write tests
- Implement
- Profile

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface
Code
Build
Debug
Write tests
Implement
Profile
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Code
- Build
- Debug
- Write tests
- Profile

Implement
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Code
- Build
- Debug
- Write tests
- Implement
- Profile

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code
Build
Debug
Write tests
Implement
Profile
Start with a project
Start with a project

- Xcode is project based (most of the time)
Start with a project

- Xcode is project based (most of the time)
- Start an application with a project template
Start with a project

• Xcode is project based (most of the time)
• Start an application with a project template
• What is a project?
  ▪ Collection of source code files (or resources)
Start with a project

• Xcode is project based (most of the time)
• Start an application with a project template
• What is a project?
  ▪ Collection of source code files (or resources)
  ▪ Definition of what that source code turns into (a product)
Start with a project

• Xcode is project based (most of the time)
• Start an application with a project template
• What is a project?
  ▪ Collection of source code files (or resources)
  ▪ Definition of what that source code turns into (a product)
  ▪ Set of rules of how to get these
Parts of a project
Parts of a project

Source Code
(resources)
Parts of a project

Source Code (resources)

Product
Parts of a project

Source Code (resources)

Target

Product
Creating core classes

• Class creation is easy
• Create a new file from the templates and give it a name
• Done.
Building an interface
Building an interface

- You use the most obvious tool: Interface Builder.
Building an interface

• You use the most obvious tool: Interface Builder.
• Interface Builder is a drag and drop WYSIWYG editor for interfaces.
Building an interface

• You use the most obvious tool: Interface Builder.
• Interface Builder is a drag and drop WYSIWYG editor for interfaces.
• IB is not a code generator and aims to minimize the amount of code you need to write.
Building an interface

• You use the most obvious tool: Interface Builder.
• Interface Builder is a drag and drop WYSIWYG editor for interfaces.
• IB is not a code generator and aims to minimize the amount of code you need to write.
• IB provides mechanisms to connect your code to your interface (and vice versa) called connections.
Software Development Cycle
And how the tools integrate with it

- Start with a project
- Make core classes
- Design an interface
- Code
- Build
- Debug
- Write tests
- Implement
- Profile
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code — Build — Debug — Write tests — Implement

Profile
Implementation — Write code
Implementation — Write code

• Xcode is a powerful editor.
Implementation — Write code

• Xcode is a powerful editor.
• Does expected things like line numbers and color syntax.
Implementation — Write code

• Xcode is a powerful editor.
• Does expected things like line numbers and color syntax.
• Also does “smart” things:
  ▪ Code focus
  ▪ Code folding
  ▪ Code sense
  ▪ Code completion
  ▪ Code refactoring
Implementation — Write code

• Navigation cheat sheet
## Implementation — Write code

- Navigation cheat sheet

<table>
<thead>
<tr>
<th>Open Quickly</th>
<th>Command-Shift-D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementation — Write code

- Navigation cheat sheet

<table>
<thead>
<tr>
<th>Open Quickly</th>
<th>Command-Shift-D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementation — Write code

- Navigation cheat sheet

<table>
<thead>
<tr>
<th>Open Quickly</th>
<th>Command-Shift-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch between interface and implementation</td>
<td>Command-Option-Up-Arrow</td>
</tr>
</tbody>
</table>
Implementation — Write code

- Navigation cheat sheet

<table>
<thead>
<tr>
<th>Open Quickly</th>
<th>Command-Shift-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch between interface and implementation</td>
<td>Command-Option-Up-Arrow</td>
</tr>
<tr>
<td>Jump to definition of symbol</td>
<td>Command-Double-Click</td>
</tr>
</tbody>
</table>
## Implementation — Write code

- **Navigation cheat sheet**

<table>
<thead>
<tr>
<th>Action</th>
<th>keyboard shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Quickly</td>
<td>Command-Shift-D</td>
</tr>
<tr>
<td>Switch between interface and implementation</td>
<td>Command-Option-Up-Arrow</td>
</tr>
<tr>
<td>Jump to definition of symbol</td>
<td>Command-Double-Click</td>
</tr>
<tr>
<td>Jump to documentation of symbol</td>
<td>Option-Double-Click</td>
</tr>
</tbody>
</table>
In Person.m
#import "Occupations.h"

@implementation Person
...
@end

In Occupations.h
#import "Programmer.h"
#import "Mailman.h"
#import "Fireman.h"
...

Saturday, 7 February 2009
Implementation — Build

- This is what the target does.

In Person.m

```c
#import “Occupations.h”

@implementation Person
...
@end
```

In Occupations.h

```c
#import “Programmer.h”
#import “Mailman.h”
#import “Fireman.h”
...
```
Implementation — Build

• This is what the target does.
• Xcode’s build system figures out dependencies for you, and what needs to be built.

In Person.m

```
#import "Occupations.h"

@implementation Person
...
@end
```

In Occupations.h

```
#import "Programmer.h"
#import "Mailman.h"
#import "Fireman.h"
...
```
Implementation — Build

• Drops down to gcc and ld for compiling and linking.

• Configurations
  ▪ A group of settings to send to gcc and ld for a target.
  ▪ Default configurations are Debug and Release
    ▪ **Debug**: Has standard settings with debug symbols. Only setting you always need to change is to turn on warnings as errors.
    ▪ **Release**: Same as Debug, but without debug symbols.
Implementation — Debug
Implementation — Debug

• Xcode sits on top of GDB for debugging, and brings it into the developer’s workspace.
Implementation — Debug

• Xcode sits on top of GDB for debugging, and brings it into the developer’s workspace.

• In-line debugging:
  ▪ Roll over variables to see their values.
  ▪ Step through your code without leaving it.
Implementation — Debug

• Xcode sits on top of GDB for debugging, and brings it into the developer’s workspace.

• In-line debugging:
  ▪ Roll over variables to see their values.
  ▪ Step through your code without leaving it.

• Breakpoints:
  ▪ To set them, just click on the source line you want to stop at.
  ▪ Use Breakpoints window to set conditional breakpoints.

• **Tip:** Always break on objc_exception_throw.
“Just because Xcode sits on top of gcc, ld, and gdb, this does not give you an excuse not to know them.”

Useful links:
http://developer.apple.com
http://users.ece.utexas.edu/~adnan/gdb-refcard.pdf
Implementation — Testing

• Xcode includes SenTest, an Objective-C and C/C++ unit testing framework.

• Create a SenTest target, and then set that target as a dependency of your others.

• If your unit tests fail, then your build fails, and you can quickly find the problem.

• Read Apple’s documentation for more.

• iPhone SDK currently does not support SenTest.
Demo
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code Build Debug Write tests Implement
Profile

Saturday, 7 February 2009
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface

Code → Build → Debug → Write tests → Implement

Profile
Profiling
Profiling

• Instruments rocks.
Profiling

• Instruments rocks.
• You can profile all different aspects of your program:
  ▪ CPU sampler (where are you spending the most time)
  ▪ Memory allocations
  ▪ Leaks
  ▪ File system usage
  ▪ Network usage
  ▪ Graphics performance
Profiling

• From Xcode, instead of debugging, you run with a template from Instruments, or build your own.
Software Development Cycle
And how the tools integrate with it

Start with a project
Make core classes
Design an interface
Code
Build
Debug
Write tests
Implement
Profile
Summary
Summary

• Xcode is a…
Summary

• Xcode is a...
  • Project manager
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
  ▪ Build system
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
  ▪ Build system
  ▪ Debugging environment
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
  ▪ Build system
  ▪ Debugging environment

• Interface Builder is an…
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
  ▪ Build system
  ▪ Debugging environment

• Interface Builder is an…
  ▪ Interface builder
Summary

• Xcode is a…
  ▪ Project manager
  ▪ Editor
  ▪ Build system
  ▪ Debugging environment

• Interface Builder is an…
  ▪ Interface builder

• Instruments is an…
Summary

- Xcode is a...
  - Project manager
  - Editor
  - Build system
  - Debugging environment
- Interface Builder is an...
  - Interface builder
- Instruments is an...
  - Application profiler
Introduction to Apple’s Developer Tools

The Xcode tools in the software development cycle
Using Xcode to write code
Using Xcode to debug and test code
Introduction to Interface Builder
Introduction to profiling code with Instruments

acm.uiuc.edu/macwarriors/devphone
About These Slides

If you need to reference the content of this presentation, these slides will be posted on our website, listed below:

acm.uiuc.edu/macwarriors/devphone
macwarriors