Part I

Learning to Program

COPYRIGHTED MATERIAL

In This Part

CHAPTER 1

Programming for World of Warcraft

World of Warcraft (WoW) was released Nov. 23, 2004, and very quickly became *the* model for Massively Multiplayer Online Role Playing Games (MMORPG). Providing an intuitive user interface and a low barrier to success the game is currently played by more than 11 million users, including their friends, co-workers, and families. WoW has something enjoyable for those players who spend six hours a night raiding with their guilds, the cubicle warriors who play for half an hour a day on their lunch breaks, and a large range of individuals in between.

Beyond killing monsters and questing for glory, there is another side to World of Warcraft, a game within a game. Blizzard has provided an extremely powerful system for creating third-party addons and writing macros, and users have been taking advantage of the open system since the beta test for the game. This book is designed to introduce you to the world of customizing World of Warcraft and show you how to create custom addons.

Customizing the User Interface

The World of Warcraft game client consists of two major parts: the game world and the user interface. The game world is the three-dimensional space in which your character resides. This includes the buildings and terrain, other players and enemies, and interactive objects such as herbs, mining veins, mailboxes, and signposts. The game world also includes some non–three-dimensional objects, namely the character names and titles, and the numbers that show the damage your character has done. These elements are not accessible through the scripting interface and cannot be modified.

The user interface comprises the other elements in the client, including the action buttons, unit frames, maps, and options windows. Addons can be written to add or modify existing elements to add functionality or to show information in a different way.

What Is an Addon?

An *addon* is a collection of files inside a named directory within the World of Warcraft directory. These files are loaded by the game's scripting system and executed within the client to make some modification to the user interface. This definition of addons does not include any third-party executables that are run outside the game (those sorts of programs are normally prohibited by WoW's terms of service).

The average addon consists of individual components that work together to create a final product, possibly including:

- A table of contents file that identifies the addon and the files to be loaded
- Media files, such as graphics and sounds
- Lua scripts that define the behavior of the addon
- XML files that define the visual elements of the addon

The first part of this book is designed to introduce you to the Lua programming language and the XML markup that is specific to World of Warcraft. These skills are an important part of writing addons effectively. If you are already proficient in Lua and XML, you can skip ahead to Part II of the book, which covers the use of the World of Warcraft API in creating addons; however, you will likely find the material in Part I worthwhile.

What Can Addons Do?

Addons typically fall into one or more of the following categories:

- Displaying additional information, such as the sale price of quest rewards (Figure 1-1), or approximately how many more of a given spell you can cast without running out of power (Figure 1-2).
- Changing the display of interface elements, such as the combat text information (Figure 1-3), or making the auction house interface easier to navigate (Figure 1-4).
- Providing new ways for the player to take action (targeting units, casting spells) within the game, such as replacement unit frames (Figure 1-5) or alternate action buttons (Figure 1-6).

Figure 1-1: Valuation showing sell price for items

Figure 1-2: Dr. Damage displaying number of possible casts

Figure 1-3: MikScrollingBattleText displaying combat information

Figure 1-4: Auctioneer displaying auction listings in a compact form

Figure 1-5: Grid unit frames showing the status of a raid

Figure 1-6: Bartender4 with ButtonFacade_Serenity providing alternate action buttons

Prior to the release of the Burning Crusade expansion pack to World of Warcraft, there were several addons that Blizzard deemed against the spirit and intention of the game. These addons were later disabled and changes were

made to the scripting system to prevent their use. As a result the following actions are unavailable to addons:

- Automatic character movement
- Automatic target selection
- Automatic selection and use of spells or items
- Real-time communication with external programs

In the past, Blizzard has been asked about the limits of the scripting/macro system. Its response has been that it is interested in ''smart players,'' not ''smart buttons.'' In other words, addons and macros can work to display information to the users or allow them to access functionality in an easier way, but should not be used to make automatic decisions.

In addition, addons are forbidden from doing anything that would otherwise be against the World of Warcraft ''Terms of Use,'' which you can find at `http://worldofwarcraft.com/legal/termsofuse.html`.

Exploring Your AddOns Directory

As mentioned previously, all addons must exist within a subdirectory under your World of Warcraft directory. Depending on what operating system you are using and how you have installed the game, this directory may exist in one of a few places (see Table 1-1). If you happen to be running Windows Vista, the location of your installation will depend on how the computer has been configured and where the game was installed. During the installation of Wrath of the Lich King or patch 3.0.2 you should have been asked to move the game to option #3. If you agreed to this change, you may have two versions of World of Warcraft, with the old one not being used any more.

Table 1-1: Default World of Warcraft Installation Directory

OPERATING SYSTEM	DEFAULT INSTALLATION DIRECTORY
Microsoft Windows 98, 2000, or XP	`C:\Program Files\World of Warcraft`
Microsoft Windows Vista (option #1)	`C:\Program Files\World of Warcraft`
Microsoft Windows Vista (option #2)	`C:\Users\<username>\AppData\Local\VirtualStore\Program Files\World of Warcraft`
Microsoft Windows Vista (option #3)	`C:\Users\Public\Games\World of Warcraft`
Mac OS X	`/Applications/World of Warcraft`

If you have launched World of Warcraft previously, there should be an Interface directory within and an `AddOns` directory below that. This is where all addons are stored.

Blizzard Addons

Much of the functionality in the default user interface is implemented via modular addons that are loaded only when needed by the user. When the player visits an auctioneer, for instance, the game loads the `Blizzard_AuctionUI` addon.

Having the addons in separate load-on-demand modules allows addon authors to easily override the default functionality (such as replacing the auction house interface rather than just changing it). In addition, the modularity speeds up load times when starting the game. Table 1-2 describes the existing Blizzard addons.

Table 1-2: Blizzard Load-on-Demand Addons

ADDON NAME	PURPOSE
`Blizzard_AchievementUI`	Explore the achievements your character can complete and those he has already completed.
`Blizzard_ArenaUI`	Display unit frames for enemy units in arena PVP.
`Blizzard_AuctionUI`	Search for items available for sale, as well as posting new items up for auction.
`Blizzard_BarbershopUI`	Customize the facial features and hair style/color for your character.
`Blizzard_BattlefieldMinimap`	Display a smaller version of the world map, including the PVP objectives.
`Blizzard_BindingUI`	Customize the keyboard bindings made available by the default and custom interfaces.
`Blizzard_Calendar`	Display a calendar that shows the various scheduled game events and allows players to create their own events.
`Blizzard_CombatLog`	Present combat information in a linear combat log that can be filtered and colored via options.

Continued

Table 1-2: (*continued*)

ADDON NAME	PURPOSE
`Blizzard_CombatText`	Show various combat events in moving text in the user interface, customizable via the options screens.
`Blizzard_DebugTools`	Provide slash commands and utility functions that are useful to addon developers.
`Blizzard_GMChatUI`	Provide a chat window for communication with game masters.
`Blizzard_GMSurveyUI`	Allow the user to fill out a survey that has been sent by Blizzard following a GM interaction.
`Blizzard_GlyphUI`	Inscribe glyphs into your spellbook in order to customize your spells.
`Blizzard_GuildBankUI`	Add and remove items and gold from your guild's bank.
`Blizzard_InspectUI`	Inspect another player to view his equipment, combat stats, and talents.
`Blizzard_ItemSocketingUI`	Socket gems into an item.
`Blizzard_MacroUI`	Edit global and character-specific macros.
`Blizzard_RaidUI`	Display unit frames for the members in your raid.
`Blizzard_TalentUI`	Assign talent points and explore the various talent trees.
`Blizzard_TimeManager`	Show a clock on the minimap and provide a simple in-game timer.
`Blizzard_TokenUI`	View the various currency tokens that your character has earned.
`Blizzard_TradeSkillUI`	Explore the various recipes that are associated with a given tradeskill.
`Blizzard_TrainerUI`	Purchase skills available from a trainer.

Each of these directories contains a single file that has the addon's name and a `.pub` extension. As far as we can tell, this is some sort of signature used by the game to verify the authenticity of the addon. Addons that are written by Blizzard are given a special ''secure'' flag that allows them to take

protected actions, something that is covered in Chapter 8. The code for the addons is actually stored in the data files for the game and can't be directly replaced.

Custom Addons

If you have downloaded any custom addons they will sit alongside the Blizzard addons in your `Interface\AddOns` directory inside subdirectories. Unlike the official addons, these addon directories actually contain the files that are necessary to load and run the addon. The organization and contents of these files varies depending on the addon. Each author has his or her own preferences and style and these differences are reflected in the way the addon is packaged and the way the code is written. Although we provide some recommendations for writing and packaging your addons, you are free to develop a style that works best for you.

Creating Your First Addon: HeyThere

Before you delve into Lua and XML, take a look at a very simple addon example so you'll have an idea of how the system works. To complete the example you need to know how to create a new directory on your computer. You also need to be familiar with a text editor that saves files without special formatting. On Windows, for example, you could use Notepad to edit files; on Mac OS X, the built-in Text Editor program is sufficient.

Creating Files and Directories

First create a new directory that will contain the addon. Navigate to your `Interface\AddOns` directory and create a new directory inside called `HeyThere`.

Open your text editor and type the following into the file:

```
## Interface: 30300
## Title: Hey There!
## Notes: Provides slash commands to greet other players

HeyThere.lua
```

Save this file in the new directory as `HeyThere.toc`. Open a new file in the editor and add the following:

```
SLASH_HEYTHERE1 = "/hey"
SLASH_HEYTHERE2 = "/heythere"
SlashCmdList["HEYTHERE"] = function(self, txt)
  if UnitExists("target") then
```

```
    SendChatMessage("Hello " .. UnitName("target"), "SAY")
  else
    SendChatMessage("Hey there everybody!")
  end
end
```

Save this file as `HeyThere.lua` in the same directory and close the text editor. Don't worry right now about what any of this code does; it's just an example addon to get you familiar with creating files and directories. You'll learn what the code does later in the book.

Loading and Testing the Addon

If you have World of Warcraft open, you must close it so it can recognize the new addon. Once you've re-opened the game client, log in to your account and stop at the character selection screen. In the bottom left of the screen should be a button named AddOns. Click it and a window similar to one shown in Figure 1-7 opens. The window shows that WoW recognizes your addon and will try to load it if it is enabled.

TIP **You may find it useful to create a character on a server that is different from your main server for addon development. This allows you to easily change which addons are enabled and disabled without affecting the characters with which you normally play.**

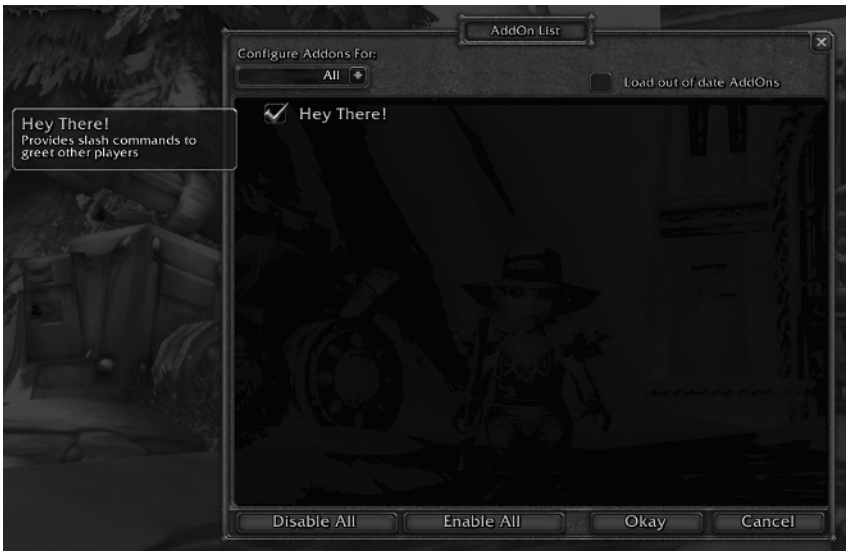

Figure 1-7: Addon selection screen showing your new addon

Ensure that the addon is enabled by checking the box to the left of the addon name. Click Okay to exit the addon selection screen and enter the game.

This addon adds two new slash commands that allow you to greet people in the world. You can type either **/heythere** or simply **/hey** and depending on whether you have something targeted your character will display one of two messages (see Figure 1-8).

Figure 1-8: HeyThere greeting with (left) and without (right) a target

If for some reason you do not see the addon in the addon selection list, ensure that you've created the files and directories correctly. The layout should be as follows:

- `Interface\AddOns\HeyThere`
- `Interface\AddOns\HeyThere\HeyThere.toc`
- `Interface\AddOns\HeyThere\HeyThere.lua`

If you get an error or have any other issues, double-check that you've typed everything correctly in each of the files. Alternatively, download the addon from this chapter's section of the website at `http://wowprogramming.com/chapters/01` and compare it to the version you have created.

Summary

This chapter introduced you to the addon system for World of Warcraft. The specific limitations and capabilities of the system were listed, along with a description of the addons that Blizzard has included with the game. You created your first addon and tested in-game to ensure it worked correctly.

Chapter 2 introduces you to the basics of the Lua programming language, used extensively when creating addons.