Object Tcl

Introduction

This page provides a simple example of how to:

- Export C++ classes to the Object Tcl system
- Describe new classes in Object Tcl
- Inherit Object Tcl classes from C++ classes
- Build an augmented Tcl shell
- Manipulate Object Tcl objects

C++ Class

Class `A` is a simple C++ class that maintains a reference (pointer) to another class `A` object and provides three methods. The first method, `doIt`, is a virtual method that prints a simple message identifying itself as class `A`'s version of `doIt`. The second method, `doItNext`, invokes the `doIt` method upon the other class `A` object referenced by the private attribute `next`. The final method, `setNext`, can be used to set the `next` class `A` object reference.

Class `A` is described and implemented in C++ in the following two files:

File A.H

```
#ifndef A_H
#define A_H

// File A.H

class A
{
public:
   A (A *next);                  // Constructor member function
   virtual  A ();                // Destructor member function
   virtual void doIt (void);     // Virtual member function
   void doItNext (void);         // Member function
   void setNext (A *obj);        // Member function
private:
   A *next;                      // Member variable
};
```

```
#endif // A_H
```

File A.C

```
#include <iostream.h>
#include "A.H"

// File A.C

A::A (A *n)
{
   next = n;
   cout << "A::constructed" << endl;
}

A:: A ()
{
   cout << "A::destructed" << endl;
}

void A::doIt (void)
{
   cout << "A::doIt" << endl;
}

void A::setNext (A *n)
{
   next = n;
}

void A::doItNext (void)
{
   if (next != NULL)
   {
      next->doIt();      // "doIt" is dynamically bound so it could be the
                         // "doIt" of class A or a subclass of A.
   }
}
```

Exporting C++ Class A to Object Tcl

The following file contains the CDL for class `A`. This file may be processed by the CDL processor and the resulting C++ files linked into the final application.

```
# File A_cdl.cdl

pass {
// Generated from A_cdl.cdl
#include "A.H"
```

```
#include "A_cdl.H"
}

class A {
   constructor {obref A}
   method doIt -dynamic {void} {void}   # "doIt" required dynamic binding
   method doItNext -static {void} {void}
   method setNext -static {obptr A} {void}
   # The "A" after the obref type indicates the actual type of the
   # parameter expected by the "setNext" method.
}
```

An Object Tcl Class

Class `B` is a new class described in Object Tcl, below, that inherits from class `A`.

Class `B` provides its own version of the `doIt` method, that is also defined in class `A`. Class `B`'s version of the `doIt` method prints a simple message identifying itself and then calls the `doIt` method of its parent class (`A`).

```
# File B.tcl

otclInterface B -isA A {
   constructor {next value}
   method doIt {}       # Redefines the "doIt" method available in class A
}

otclImplementation B {
   # The constructor method passes on its first argument up to
   # the constructor of the "A" parent class.
   constructor {n v} {{A $n}} {
      set value $v
      puts "B::constructed with value $value"
   }
   destructor {
      puts "B::destructed"
   }
   # The new version of the "doIt" method.
   method doIt {} {
      puts "B::doIt, value is $value, calling A::doIt"

      # Invoke the "doIt" method but make sure it is the version
      # of the "A" parent class and this version that would be
      # chosen by default using dynamic binding.
      $this -A doIt
   }
   attribute value
}
```

Building

To build a version of the Tcl shell (tclsh) that includes the Object Tcl package, with class `A` and class `A` exported to Object Tcl, perform the following:

1. Compile class `A`:

   ```
   system% CC -c A.C
   ```

2. Process the CDL for class `A`:

   ```
   system% cdl -h A_cdl.cdl A_cdl.H
   system% cdl -s A_cdl.cdl A_cdl.C
   ```

3. Compile the C++ code generated by the processing of `A`'s CDL description:

   ```
   system% CC -I$(OTCL) -I$(TCL) -c A_cdl.C
   ```

4. Link the object files together using a C++ linker:

   ```
   system% CC -o examplesh -L$(OTCL) -L$(TCL) A.o A_cdl.o \
   $(OTCL)/tclAppInit.o -lotcl -ltcl -lm
   ```

Note: the files `tclAppInit.o` and `tclMain.o` are modified version of the files from the tcl7.3 distribution. These files have been modified to initialize the Object Tcl package and compile under C++.

Testing

To test the augmented Tcl shell:

```
system% ./examplesh
% source TestB.tcl
About to construct an instance of class A
A::constructed

About to call method doIt on instance of class A
A::doIt

About to construct an instance of class B
A::constructed
B::constructed with value 5

About to call method doIt on instance of class B
B::doIt, value is 5, calling A::doIt
A::doIt
```

```
Setting instance of class A to point at instance of class B
About to call method doItNext on instance of class A
B::doIt, value is 5, calling A::doIt
A::doIt

About to destruct instance of class A
A::destructed

About to destruct instance of class B
B::destructed
A::destructed
%
```

Where `TestB.tcl` is:

```
# TestB.tcl

puts "About to construct an instance of class A"
set a [otclNew A ""]

puts "\nAbout to call method doIt on instance of class A"
$a doIt

source B.tcl
puts "\nAbout to construct an instance of class B"
set b [otclNew B "" 5]

puts "\nAbout to call method doIt on instance of class B"
$b doIt

puts "\nSetting instance of class A to point at instance of class B"
$a setNext $b
puts "About to call method doItNext on instance of class A"
$a doItNext

puts "\nAbout to destruct instance of class A"
otclDelete $a

puts "\nAbout to destruct instance of class B"
otclDelete $b
```

---

otcl@x.co.uk