Debugging Tutorial:

Finding and Fixing Bugs in an Application

Excerpted from *Programming for the Newton, 2nd edition* Julie McKeehan and Neil Rhodes, Academic Press. 1996

Beeping Button Brouhaha

We've written a very simple application in which the user can write a number. Pressing the "Beep" button makes the Newton beep that many times. Figure 1 shows the application. Figure 2 shows the application templates in NTK.

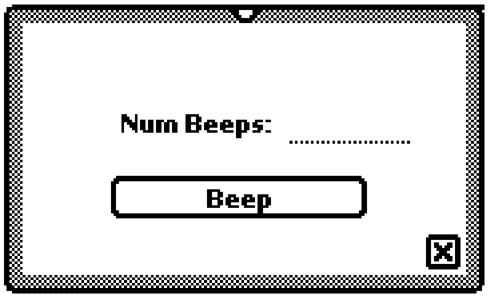
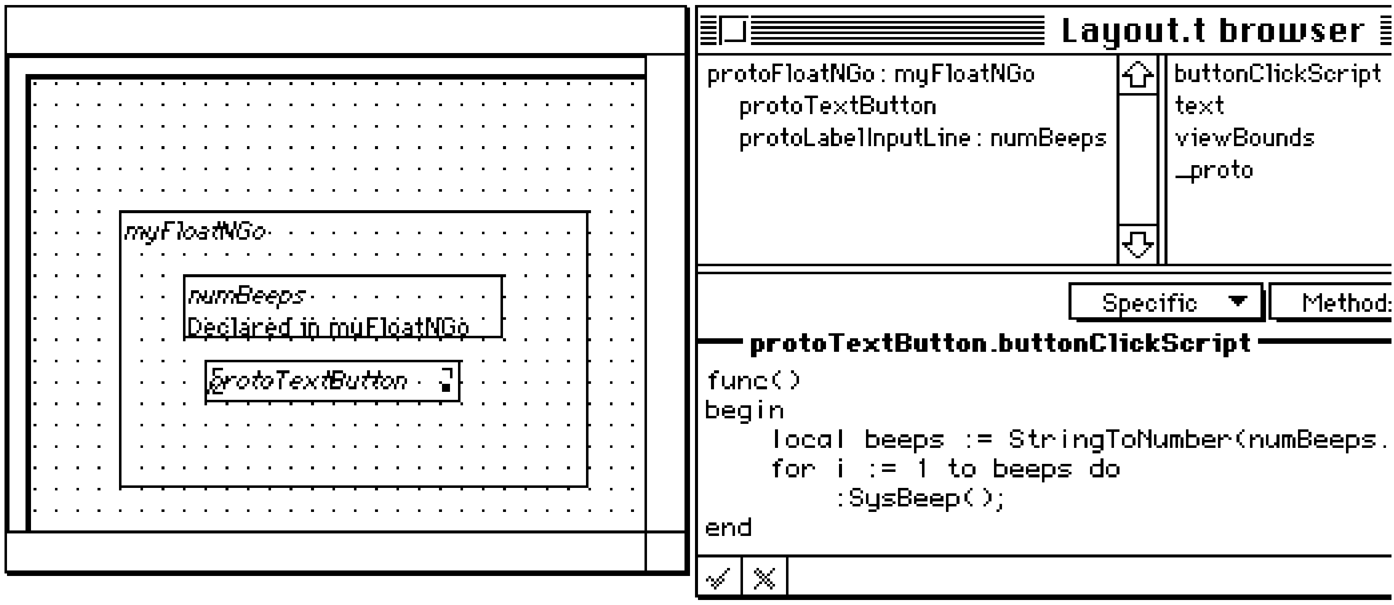


Figure 1: Beeping Button application.



```
func()
begin
    local beeps := StringToNumber(numBeeps.
    for i := 1 to beeps do
        :SysBeep();
end
```

Figure 2: Structure of the Beeping Button project.

First Problem

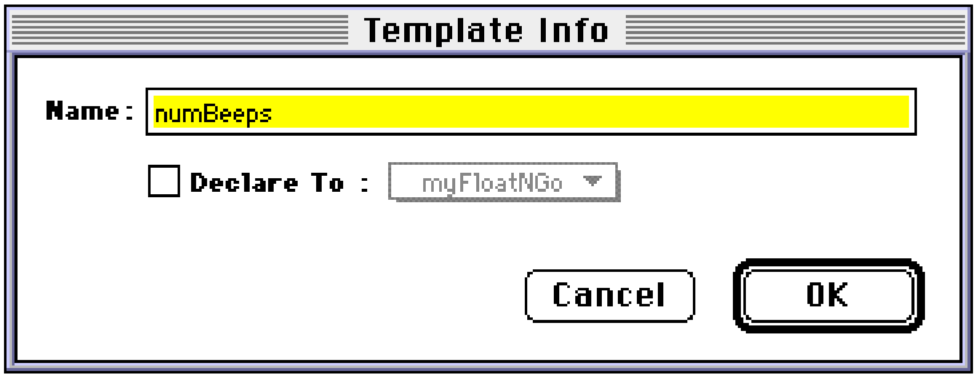
The first thing that happens when we write in a number and then press the Beep button is a notification on the Newton: “Sorry, a problem has occurred (-48807)." Let’s turn on breakOnThrows (clicking the icon in the Inspector Window) and press the button again. Now, the Inspector prints out

```
        Undefined variable: numBeeps
        evt.ex.fr.intrp;type.ref.frame
        -48807
    (#6008D1D1).buttonClickScript(), 3: Push 'text
    Entering break loop: level 1
```

If we look at the code we see that we’re trying to access the numBeeps variable from our buttonClickScript. That variable should refer to our protolabelInputLine view. What could be wrong? If we’ve correctly declared numBeeps to the protoFloatNGo, the protoFloatNGo view should have a slot named numBeeps. Let’s look in that view for numBeeps. (That view is the parent of our current self.).First, let’s get self:

```
    // get current self
    beepingButton := GetCurrentReceiver(0);

    #440F63D {_parent: {_parent: {#440D221},
                        _proto: {#6008D0C1},
                        viewCObject: 0x110926D,
                        viewclipper: 17863292,
                        base: <1>,
                        viewFlags: 577,
                        viewBounds: {#440F5C1}},
                _proto: {buttonClickScript:<function, 0
     arg(s)#6008D239>,
                         text: "Beep",
                         viewBounds: {#6008D539},
                         _proto: {@226}},
                viewCObject: 0x110A530,
                viewFlags: 515}
```
Now, let’s get the parent slot and we will have the right view:
```
     floatNGo := beepingButton._parent
     #440F5DD  {_parent: {minute: 178,
                          downButton: {#440B2E9},
                          calculator: {#4406159},
                          mailEditor: {#44064C1},
                          extrasDrawer: {#4409671},
                          defaultTransport:Newton:
     {#4405DD9},
                          OutOfMemoryAlert: {#4405D95},
                          notification: {#4405D35},
                          remindSlip: {#44060C5},
                          namesButton: {#44063D9},
                          folderEdit: {#4405DF1},
                          phoneKeyboard: {#4405ECD},
                          ovButton: {#440643D},
                          upButton: {#4406461},
                          thegang: {#44065F9},
                          printerSerialPicker: {#4405D05},
                          ...},
               _proto: {viewBounds: {#6008D199},
                         stepChildren: [#6008D1B9],
                         _proto: {@180},
                         debug: "myFloatNGo",
                         appSymbol: |Demo:NTK.Demo|},
               viewCObject: 0x110926D,
               viewclipper: 17863292,
               base: <1>,
               viewFlags: 577,
               viewBounds:{left:-25, top:173, right:139,
     bottom:265}}
```
The numBeeps slot doesn’t seem to be in the floatNGo. The view otherwise appears to be correct. It sounds like a problem in declaring. Let’s check the Template Info dialog for that template (see Figure 3). Well, well, well. Turns out it actually wasn’t declared. We’ll checkmark the “Declare To:” checkbox and rebuild.



Name: numBeeps

Figure 3: Template Info dialog showing undeclared numBeeps.

Second Problem (a Hard One)

We rebuild, download, and rerun. We write in a number and tap the "Beep" checkbox. We get the following error in the Inspector:

```
    Undefined variable: numBeeps
    evt.ex.fr.intrp;type.ref.frame
    -48807
(#6008D229).buttonClickScript(), 3: Push 'text
Entering break loop: level 1
```

This is exactly the same error at the same place we had it before. Let's check the `floatNGo` view again:

```
     beepingButton := GetCurrentReceiver(0);
     floatNGo := beepingButton._parent
```

and here is what we find when the Inspector returns our result:

```
     #4412B0D  {_parent: {minute: 181,
                          downButton: {#440B129},
                          calculator: {#4406159},
                          mailEditor: {#44064A1},
                          extrasDrawer: {#4409651},
                          defaultTransport:Newton:
     {#4405DD9},
                          OutOfMemoryAlert: {#4405D95},
                          notification: {#4405D35},
                          remindSlip: {#44060C5},
                          namesButton: {#44063D9},
                          folderEdit: {#4405DF1},
                          phoneKeyboard: {#4405ECD},
                          ovButton: {#440643D},
                          upButton: {#440EF4D},
                          thegang: {#44065D9},
                          printerSerialPicker: {#4405D05},
                          ...},
                _proto: {viewBounds: {#6008D1F1},
                         stepChildren: [#6008D211],
                         _proto: {@180},
                         debug: "myFloatNGo",
                         numBeeps : NIL,
                         stepAllocateContext: [#6008D731],
                             appSymbol: |Demo:NTK.Demo|},
                  viewCObject: 0x1108C2C,
                  numBeeps : {_parent: <2>,
                               _proto: {#6008D5D1},
                               viewCObject: 0x1109F0C,
                               entryLine: {#4419229},
                               labelLine: {#4418E49},
                               width: 73,
                               indent: 75,
                               height: 13},
                  viewclipper: 17863746,
                  base: <1>,
                  viewFlags: 577}
```

The numBeeps slot seems to be there and seems to point to what looks like it could be our view. Let’s try to access it from the Inspector:

```
floatNGo.numBeeps
#2        NIL
```

That doesn’t make sense. We can see that it is there. Let’s try another way to get to that view using the Debug function:

```
Debug("numBeeps")
#2        NIL
```

Curiouser and curiouser. However, look very closely at the way the numBeeps slot prints out versus any other slot:

```
_proto: {viewBounds: {#6008D1F1},
                   stepChildren: [#6008D211],
                   _proto: {@180},
                   debug: "myFloatNGo",
                   numBeeps : NIL,
                   stepAllocateContext: [#6008D731],
                   appSymbol: |Demo:NTK.Demo|},
          viewCObject: 0x1108C2C,
          numBeeps : {_parent: <2>,
                       _proto: {#6008D5D1},
```

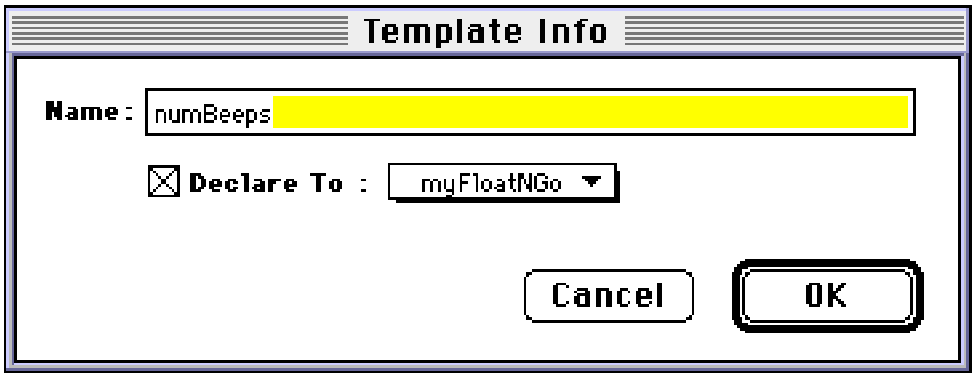
Other slots have no space before the colon (“:”), while the numBeeps slot has one space there. Could this have anything to do with our problem? What if that space were significant? Let’s try calling Debug with an extra space after numBeeps:

```
Debug("numBeeps ")
```

and here is the Inspector return result that we get:

```
#4418AF5  {_parent: {_parent: {#4412B25},
                      _proto: {#6008D0C1},
                      viewCObject: 0x1108C2C,
                      numBeeps : <2>,
                      viewclipper: 17863746,
                      base: <1>,
                      viewFlags: 577},
          _proto: {viewBounds: {#6008D681},
                   label: "Num Beeps:",
                   entryFlags: 10753,
                   _proto: {@189},
                   debug: "numBeeps ",
                         preAllocatedContext: |numBeeps |},
                viewCObject: 0x1109F0C,
                entryLine: {_parent: <2>,
                            _proto: {#356429},
                            viewCObject: 0x110A83B,
                            viewFlags: 10753,
                            viewBounds: {#4418F4D},
                            text: "2"},
                labelLine: {_parent: <2>,
                            _proto: {#356569},
                            viewCObject: 0x110A871,
                            text: "?Num Beeps:",
                            viewFont: {@100},
                            viewBounds: {#4418E2D}},
                width: 73,
                indent: 75,
                height: 13}
```

So if it acts as though the name had an extra space—maybe it does. Let’s check the Template Info dialog for that template more carefully (see Figure 4). Indeed, there is a trailing space after numBeeps. We’ll delete it and rebuild.


Template Info

Name: numBeeps

☒ Declare To : myFloatNGo

Figure 4: Template Info dialog for numBeeps with an extra space at the end.

Third Problem

We rebuild, download, and rerun. We write the number “2” and tap “Beep." We get the following error in the Inspector:

```
Expected an integer, got nil
evt.ex.fr.type;type.ref.frame
-48406
(#6008D229).buttonClickScript(), 27: PushConstant NIL
Entering break loop: level 1
```

We’re still in our buttonClickScript, about to execute the code at program counter 27. Here’s the NewtonScript code for the buttonClickScript:

```
func()
begin
     local beeps := StringToNumber(numBeeps.text);
     for i := 1 to beeps do
```

```
                :SysBeep();
     end
```

Let’s look at the disassembled code for the `buttonClickScript`:

```
     Disasm(GetCurrentFunction(0))
         0: FindVar              numBeeps
         1: Push                 'text
         2: GetPath              1
         3: Push                 'StringToNumber
         4: Call                 1
         5: SetVar               beeps
         6: PushConstant         1
         7: SetVar               i
         8: GetVar               beeps
         9: SetVar               i|limit
        10: PushConstant         1
        11: SetVar               i|incr
        12: GetVar               i|incr
        13: GetVar               i
        14: Branch               23
        17: PushSelf
        18: Push                 'SysBeep
        19: Send                 0
        20: Pop
        21: GetVar               i|incr
        22: IncrVar              i
        23: GetVar               i|limit
        24: BranchIfLoopNotDone  17
        27: PushConstant         NIL
        28: Return
     #2         NIL
```

We’re about to execute the code at offset 27—it looks as though we’re at the end of the loop. Let’s look at the values of our variables to see if they are reasonable:

```
     GetAllNamedVars(0);
     #4419641  {beeps: NIL,
                i: 1,
                i|limit: NIL,
                i|incr: 1,
                numBeeps: {_parent: {#4418345},
                           _proto: {#6008D5D9},
                           viewCObject: 0x110A57D,
                           entryLine: {#44186F5},
                           labelLine: {#44181FD},
                           width: 73,
                           indent: 75,
                           height: 13},
                text: "Beep",
                SysBeep: <function, 0 arg(s) #350E8D>}
```

It doesn’t seem right that `beeps` is nil. In fact, that would explain the error we got. The for loop expected an integer, but got nil. But why is `beeps` nil? It was obtained from calling `StringToNumber` on `numBeeps.text`. Let’s start by looking at `numBeeps` more closely:

```
numBeeps := GetNamedVar(0, 'numBeeps);
#4417EA9  {_parent: {_parent: {#4411D11},
                      _proto: {#6008D0C1},
                      viewCObject: 0x110A483,
                      numBeeps: <2>,
                      viewclipper: 17863765,
                      base: <1>,
                      viewFlags: 577},
           _proto: {viewBounds: {#6008D689},
                    label: "Num Beeps:",
                    entryFlags: 10753,
                    _proto: {@189},
                    debug: "numBeeps",
                    preAllocatedContext: numBeeps},
           viewCObject: 0x110A57D,
           entryLine: {_parent: <2>,
                       _proto: {#356429},
                       viewCObject: 0x110A58C,
                       viewFlags: 10753,
                       viewBounds: {#4418301},
                       text: "2"},
           labelLine: {_parent: <2>,
                       _proto: {#356569},
                       viewCObject: 0x110A5C2,
                       text: "?Num Beeps:",
                       viewFont: {@100},
                       viewBounds: {#44181E1}},
           width: 73,
           indent: 75,
           height: 13}
```

Now let's look at the value of the `text` slot in `numBeeps`. Expecting to find the value of 2, we get a surprise instead:

```
numBeeps.text
#4632AD   ""
```

There isn't a `text` slot in the `numBeeps` view or template. So the value of `numBeeps.text` must be coming from the `protoLabelInputLine` itself. Notice, however, that there does seem to be a `text` slot with the value `"2"` in `numBeeps.entryLine`. But of course! For a `protoLabelInputLine`, the input text isn't found in the text slot of the `labelInputLine` view, but in the text slot of the child view where the input is actually done. With this bit of fresh information we can now modify the code in the `buttonClickScript`. Our code should actually be

```
func()
begin
     local beeps :=
          StringToNumber(numBeeps.entryLine.text);
     for i := 1 to beeps do
          :SysBeep();
end
```

Fourth Problem

We rebuild, download, and rerun. We write in “2” in the input line and tap the Beep button. We get the following error in the Inspector:

```
Expected an integer, got <a real number>
evt.ex.fr.type;type.ref.frame
-48406
(#6008D229).buttonClickScript(), 27: PushConstant NIL
Entering break loop: level 1
```

Hmm. This is the same location where we broke before. Last time the error was “Expected an integer, got nil." This time it got a real number. Let’s take a look at beeps:

```
GetNamedVar(0, 'beeps);
#4418E79  2.00000
```

Well, that’s the problem. The beeps variable is a real number, not an integer. Using our brilliant deductive capabilities we realize that StringToNumber must return a real number. Let’s check within the Inspector:

```
StringToNumber("2");
#4419331  2.00000
StringToNumber("2.5");
#44193ED  2.50000
```

We’ll use the Floor function, which rounds down a real number to an integer, to fix our problem:

```
Floor(StringToNumber("2"));
#8       2

Floor(StringToNumber("2.5"));
#8       2
```

Let’s rewrite the buttonClickScript. We’d better keep in mind that StringToNumber might return nil if we pass in a nonnumeric string (wonder what would happen if we called Floor with nil?):

```
func()
begin
     local beeps :=
          StringToNumber(numBeeps.entryLine.text);
     if beeps then
          for i := 1 to Floor(beeps) do
               :SysBeep();
end
```

We rebuild, download, and rerun. We write in “2” in the input line and tap the Beep button. Lo and behold, the Newton beeps (although it’s hard to tell there are two beeps because the second beep starts before the first beep finishes). Just to be thorough, we write in “two” in the input line and tap the Beep button. Nothing happens, just as we desire.