

Make Widgets with Konfabulator

by Johnathon Williams



WHAT YOU NEED

- Mac OS 10.2.3 or later (\$129, www.apple.com)
- Konfabulator 1.5 or later (\$25, www.konfabulator.com)
- Image editor that supports transparency and PNG format, such as Adobe Photoshop (\$609, www.adobe.com) or GraphicConverter (\$35, www.lemkesoft.com)
- Text editor, such as BBEdit (\$179, www.barebones.com) or TextEdit (part of OS X)
- iCal with scheduled calendar of events (part of OS 10.2)



Welcome to the Widget factory. Get comfortable. You're going to want to stay awhile.

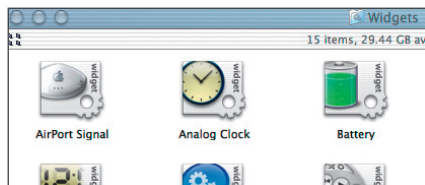
Mac addicts have a powerful new ally in their continuing war against unused leisure time. It's called Konfabulator, and it's the best thing out there if you're looking to create simple, great-looking, mini desktop applications, called *Widgets*. With a little practice, even



programming novices can use Konfabulator to create apps that do just about anything (short of snagging a date with a supermodel, cleaning the cat box, and other far-fetched ideas).

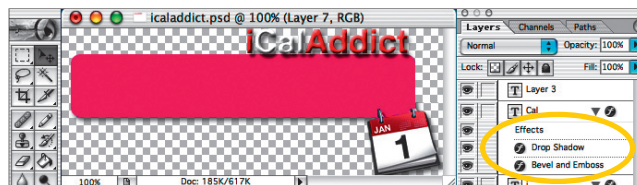
Strictly speaking, Konfabulator is billed as a "Java runtime engine" (um, no, we don't know exactly what that means either). Simply speaking, it's shareware that runs in the background, combining graphics and scripts to create Widgets for your desktop. Here, we show you how to build a simple Widget—one that displays tomorrow's first appointment from the schedule at the top of your iCal calendar list. Keep in mind that this is an introduction—take what you learn and run with it.

1 Launch Time On first launch, Konfabulator displays a series of dialogs that walk you through the initial set-up, which includes installing a bunch of premade Widgets in your Documents folder, launching a few Widgets and displaying directions on how to access and manipulate them, showing how to access the gear menu, and telling you where to find more Widgets on the Web. Once you finish the walkthrough, the app opens the Widgets folder (the one it created in your Documents folder).



Once you complete the initial setup, Konfabulator displays the Widgets it installed.

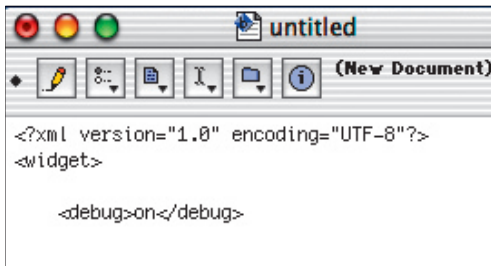
2 Make Your GUI Gorgeous Most Widgets perform a simple function, but that doesn't mean you should skip on its appearance. To create your calendar faceplate (this Widget requires only a space to display appointment info, but feel free to add other design accoutrements—we added a logo and the iCal icon to spruce up ours), launch your graphics app (we used Photoshop). First, create a new, relatively small document, keeping in mind that it will be soaking up desktop space; we set ours to 475 by 250 pixels. Next, create a new layer, turn off the Background layer's visibility, and start designing your Widget. Be sure to design at least one element that can hold two lines of text and display black text legibly. From there, layer stuff, add shadows, create bevels, play with opacity—go crazy. When finished, create a new folder called *images*, and save your work as a PNG file inside of it.



You don't need to be a skilled artist to create good-looking Widgets—Photoshop filters and a bit of copy-and-paste make for a good recipe.

3 Mark the Markup Language

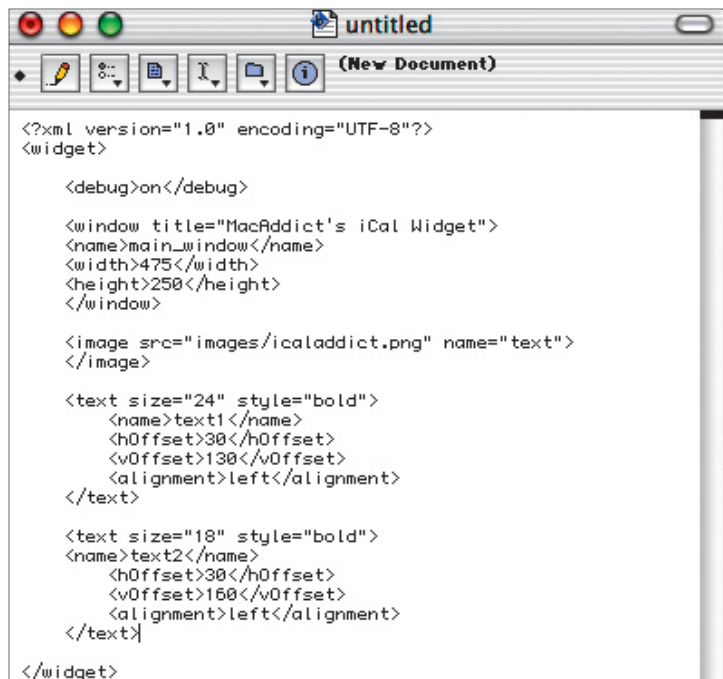
To construct a Widget, you need to feed Konfabulator display instructions via an XML file—XML (extensible markup language) is similar to HTML in that it structures information by placing it between bracketed tags. To code, start with an opening tag (for example, `<width>`), follow it with specific information for that tag (typing `400` would set the width to 400 pixels in this scenario), and then end the instruction with a closing tag (for example, `</width>`). To create our calendar Widget, launch your favorite text editor (if using TextEdit, use plain-text formatting), type `<?xml version="1.0" encoding="UTF-8"?>`, and press Return. This tells XML parsers that the file adheres to XML 1.0 specifications and is encoded in Unicode UTF-8. Then type `<widget>`, the opening tag for your Widget instruction. Press Return twice to skip a line and then press Tab to indent. Next, type `<debug>on</debug>`, press Return twice, and then press Tab—this turns on the debug menu, which helps you find problems when you test your Widget.



Before you start writing up the XML code, precede your instructions with this first line of info, so that XML parsers will know what they're dealing with.

4 Build the Skeleton with XML

Just copy our XML instructions (shown here) into your text doc, but substitute the following variables with your own: for *window title*, supply an app name; for *width* and *height*, enter your image's dimensions accordingly; for *image src*, keep the path but substitute *icaladdict.png* with your image's name. Copy the rest of the code as shown for now—you'll tweak the text offsets (`<vOffset>`) and (`<hOffset>`) later since your graphic will vary in size from ours. To keep things simple, we used only one image in our Widget, but Konfabulator can support multiple images—put each image in its own XML tag section and include offset and alignment tags for placement. You can also set opacity tags. When finished, save your file as *iCalAddict* (or the like) in plain text, but omit a file extension—you'll add one later.



The painless way to code a Widget? Plagiarize. Well, with our code anyway. Just copy our text line for line, but fill in your own variables.

A WORD ABOUT WIDGETS

Before you start building, you should know a few things about Konfabulator and Widgets in general. Because Konfabulator runs as a background app, you won't find its icon in the Dock. Instead, it presents itself as a menubar item—its icon looks like a pair of gears and is located toward the menubar's right side when it's running. To quit Konfabulator and all running Widgets, select Quit Konfabulator from this gear menu. To quit an individual Widget, Control-click the Widget interface and select Close Widget from the contextual menu.

The calendar Widget in this tutorial barely scratches the surface of what you can do with Konfabulator. Anything JavaScript and AppleScript can do, a Widget can do, so the possibilities are practically endless. One of the easiest ways to improve your Widget-making skills is to look at how others

script their Widgets. You can examine the XML and scripting of any Widget by Control-clicking a Widget's icon, selecting Show Package Contents from the contextual menu, and opening the `.kon` file in any text editor.

To expand your Widget-making abilities, we highly recommend downloading the Widget XML & JavaScript Reference file from www.konfabulator.com/workshop (if you get stuck, the site's Forums page is also a good place to seek help). This PDF contains a wealth of information about XML tags and several JavaScript extensions supported by Konfabulator. Although AppleScript knowledge is handy, to get to the meat of Konfabulator's abilities, get to know JavaScript. David Flanagan's *JavaScript, The Definitive Guide* is an excellent reference book, published by O'Reilly (this one has a Rhino on the cover).

5 Script the Brain Congratulations. You now have a perfectly valid XML skeleton for a Widget. Konfabulator could load and display this file as a Widget with no problem. However, unless you create a script that tells the Widget what to do, your GUI goodie will just sit there on your desktop, looking vacant. Though Konfabulator prefers JavaScript, it also allows you to execute AppleScript instructions from within a JavaScript. This script uses AppleScript to tap into your first calendar in iCal and copy the info for your next scheduled appointment. It starts with tomorrow's schedule and displays the info in your calendar Widget—make sure you have something scheduled before you begin. Copy the entire script shown here into the line above the closing `</widget>` tag in your XML file. Because AppleScripts must pass through JavaScript without line breaks, we added the new line indicator—`\n`—to keep commands from getting garbled. In our example, the text is wrapped for easy reading, but remember not to use the Return key when typing this script. When finished, save the file again.

```

<action trigger="onLoad">

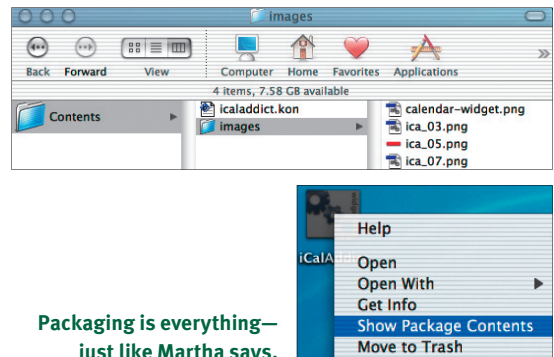
    text1.data = appleScript ('tell application "iCal" \n activate
\n set EV_1 to (first event of calendar 1 whose start date comes
after (current date)) \n get start date of EV_1 as string \n end
tell \n');

    text2.data = appleScript ('tell application "iCal" \n
get summary of (first event of calendar 1 whose start date comes
after (current date)) \n end tell \n');

</action>
</widget>
    
```

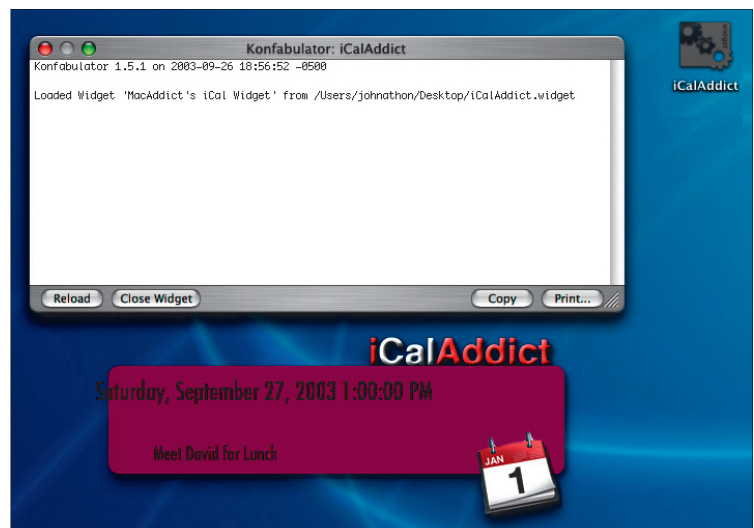
Faster than any secretary, this JavaScript-wrapped AppleScript taps into iCal and pulls data for your first appointment tomorrow.

6 Package It and Play Konfabulator stores Widgets inside of *packages*, or folders that the operating system treats as a single entity. To create a package for your Widget, first create a folder named iCalAddict (or whatever you want to name your Widget). Inside of this folder, create another folder named Contents. Drag the images folder that holds your PNG file into the Contents folder. Next, drag your XML text file into the Contents folder, and then add the extension `.kon` to your XML file. A dialog appears, asking if you really want to do this; click Use `.kon`. Now add the extension `.widget` to the iCalAddict (or equivalent) folder. Again, a dialog appears, asking if you want to make the change. Click Add, and the folder icon magically transforms into the standard Konfabulator Widget icon. To open the package, Control-click the Widget icon and select Show Package Contents from the contextual menu; a new window opens with your Contents folder inside. Close up the package by closing its window, double-click the Widget icon, and watch the fireworks.




Packaging is everything—just like Martha says.

7 Debug da Bugs But yikes! What you see is likely not what you had in mind. Relax. This is normal. Remember what we said in step 4 about the `<vOffset>` and `<hOffset>` tags—the numbers for these probably need some adjusting so that your two appointment texts align with your graphic. Along with your Widget, a debug window appears. If everything executed correctly, the debug window displays a nondescript “Loaded Widget” message. If not, you’ll see red error messages—typos are the most likely cause of errors. To fix everything, open up your Widget’s package, open your iCalAddict.kon file in your text editor, readjust the offset values, and double-check your typing. When finished, save your changes and click the Reload button at the bottom of the debug window. Repeat this process until all errors are gone. Once everything is working and looking good, go back into your iCalAddict.kon file and change the `<debug>` value from *on* to *off* to get rid of the debug window. The next time you activate your Widget, Konfabulator will display it in all its shiny, solitary glory.



Even the best programmers get bugs—time to go Orkin on them.

 Johnathon Williams is working on a Widget that will deliver a high-voltage shock to his body when he stops working and starts doing something superfluous—like Widget-making.