

Extracted from:

Pragmatic Guide to JavaScript

This PDF file contains pages extracted from Pragmatic Guide to JavaScript, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit <http://www.pragprog.com>.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printer versions; the content is otherwise identical.

Copyright © 2010 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The
Pragmatic
Programmers

Pragmatic Guide to
JavaScript

Christophe Porteneuve

Edited by David McClintock



13 Making Unobtrusive Pop-Ups

Whether you're using "actual" pop-ups (distinct windows) or pseudo pop-ups (elements on the current page, styled to look window-like), the problem remains: how do you provide access to this content for users who can't—or don't want to—open a window? Think disabled window opening, screen readers, search engines, and so on.

There's only one way: have the link *actually link to* the pop-up's contents, and apply progressive enhancement from there.

If your content is intended as an HTML fragment displayed in a pseudo pop-up (perhaps loaded through Ajax, too), you'll need to make sure you can serve it distinctly, with or without a containing document markup. That way, users accessing it through a regular link, as a stand-alone document, get something nice.

The gist of the practice here is this: link to the target content natively (`href=` and, perhaps, `target="_blank"`), and then use JavaScript to hook into these links and jazz 'em up. Baking your own window-opening code is fairly easy—it's all about the built-in `window.open()` method.

If you're adding a pseudo pop-up or lightbox, do use a good existing library and spare yourself the living hell of cross-browser issues and positioning algorithms. Here are a few pointers:

- Scripty2's UI part¹⁴
- jQuery UI¹⁵
- Dijit (based on Dojo)¹⁶
- YUI's Overlay module¹⁷
- Ext.Window¹⁸

Also keep in mind that small pop-up content may qualify for a click-triggered rich tooltip instead of a full-on pop-up zone. In such a situation, check out tooltip libraries and modules.

14. <http://github.com/madrobby/scripty2>

15. <http://jqueryui.com/>

16. <http://dojotoolkit.org/projects/dijit>

17. <http://developer.yahoo.com/yui/3/overlay/>

18. <http://www.extjs.com/deploy/dev/docs/?class=Ext.Window>

- ▶ Tag for progressive enhancement.

[Download ui/popups/index.html](#)

```
<p>
  The great thing about <a class="popup" target="_blank"
  href="http://pragprog.com/titles/pg_js">Pocket Guide to JavaScript</a>
  is that it focuses on a bunch of specific, useful tasks.</p>
```

- ▶ Script a plain `window.open()`.

[Download ui/popups/popups.js](#)

```
var POPUP_FEATURES = 'status=yes,resizable=yes,scrollbars=yes, ' +
  'width=800,height=500,left=100,top=100';

function hookPopupLink(e) {
  var trigger = e.findElement('a.popup');
  if (!trigger) return;
  e.stop(); trigger.blur();
  var wndName = trigger.readAttribute('target') ||
    ('wnd' + trigger.identify());
  window.open(trigger.href, wndName, POPUP_FEATURES).focus();
}

document.observe('click', hookPopupLink);
```

Related Tasks

- Task 12, *Pulling Off Classy Tooltips*, on page 46
- Task 15, *Creating a Lightbox Effect*, on page 52

14 Preloading Images

When your page offers user interaction that changes what images are displayed (perhaps providing a zoom, a close-up, a front/back view, or whatever), you do not want the user to see a momentary blank screen because the image you suddenly need takes an instant to load. You may want to preload such images.

In essence, there are only three ways to preload images:

- *With JavaScript, using dynamically created Image objects with appropriate src properties:* Doing so lets you detect when preloaded images are *indeed* loaded.
- *With CSS, hiding preloaded images:* This boils down to using hidden `` tags for the images to preload. You hide either the `` tags themselves or a common container (which I prefer).
- *With CSS sprites:* When you want to preload stuff like rollovers or a bunch of related images (backgrounds, borders, and corners, for instance), this should definitely be your preferred approach.¹⁹

The CSS approach is fairly straightforward, but it does not provide you with as much potential control as the JavaScript approach. The markup and script on the facing page make the assumption that image tags with a `rel="preloadZoom"` attribute have a close-up version with the same URI—with a `_closeup` name suffix—preloaded for rollover close-ups.

This is typically the kind of situation where the JavaScript option is preferable to the CSS ones. Since the rollover depends on JavaScript, we should preload in JavaScript too. (If no JavaScript is available, we won't preload stuff that can't be shown later.) Moreover, preloading in JavaScript avoids markup/styling bloat.

With JavaScript, we can avoid the risk of a temporarily blank/incomplete render by toggling to an image only when we're sure it's preloaded. The nonsprite CSS approach runs a (admittedly low) risk of toggling to an image that is not loaded yet.

For more hardcore techniques on fast image loading, take a look at Steve Souder's May 2009 presentation: *14 rules for faster-loading images*.²⁰

19. Check out the seminal <http://www.alistapart.com/articles/sprites> and the more recent <http://css-tricks.com/css-sprites/>. Steve Souders also maintains the SpriteMe tool: <http://spriteme.org/>.

20. <http://stevesouder.com/docs/wordcamp-20090530.ppt>

▶ Tag for preload.

[Download ui/preloading/index.html](#)

```

<ul id="products">
  <li><h2><a href="http://pragprog.com/titles/cppsu">
    
    <span>Prototype and script.aculo.us</span>
  </a></h2></li>
  <li><h2><a href="http://pragprog.com/titles/vsscala">
    
    <span>Programming Scala</span>
  </a></h2></li>
</ul>

```

▶ Script the preload.

[Download ui/preloading/preloading.js](#)

```

function preloadImages() {
  $$('img[rel="preloadZoom"]').each(function(img) {
    var pimg = new Image();
    pimg.src = img.src.replace(/(\.\w+$)/, '_closeup$1');
  });
}

document.observe('dom:loaded', preloadImages);

```

▶ Roll over to (ideally preloaded) images.

[Download ui/preloading/preloading.js](#)

```

function togglePreloaded(e) {
  var trigger = e.findElement('img[rel="preloadZoom"]');
  if (!trigger) return;
  if (e.type == 'mouseover') {
    trigger.src = trigger.src.replace(/(\.\w+$)/, '_closeup$1');
  } else {
    trigger.src = trigger.src.replace('_closeup', '');
  }
}

document.observe('mouseover', togglePreloaded).
  observe('mouseout', togglePreloaded);

```

The Pragmatic Bookshelf

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help you stay on top of your game.

Visit Us Online

Pragmatic Guide to JavaScript

http://pragprog.com/titles/pg_js

Source code from this book, errata, and other resources. Come give us feedback, too!

Register for Updates

<http://pragprog.com/updates>

Be notified when updates and new books become available.

Join the Community

<http://pragprog.com/community>

Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy

<http://pragprog.com/news>

Check out the latest pragmatic developments, new titles and other offerings.

Buy the Book

If you liked this eBook, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragprog.com/titles/pg_js.

Contact Us

Online Orders:	www.pragprog.com/catalog
Customer Service:	support@pragprog.com
Non-English Versions:	translations@pragprog.com
Pragmatic Teaching:	academic@pragprog.com
Author Proposals:	proposals@pragprog.com
Contact us:	1-800-699-PROG (+1 919 847 3884)