

*When Elephants Dance,  
Mice Must Be Careful:*

**Content Provider  
Conflict on the Modern  
Web**



---

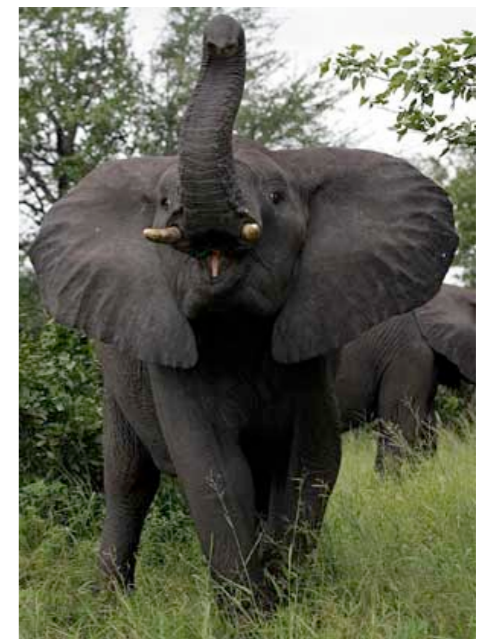
Terri Oda, Anil Somayaji, Tony White  
School of Computer Science, Carleton University  
Ottawa, Ontario, Canada

# Modern Webpages

---

Webpages get content from many sources:

- Images
- Advertisements
- Video
- Comments
- Blog “trackbacks”
- Search
- News feeds
- Chat
- Games
- Friend activity



# Combining Sources



+



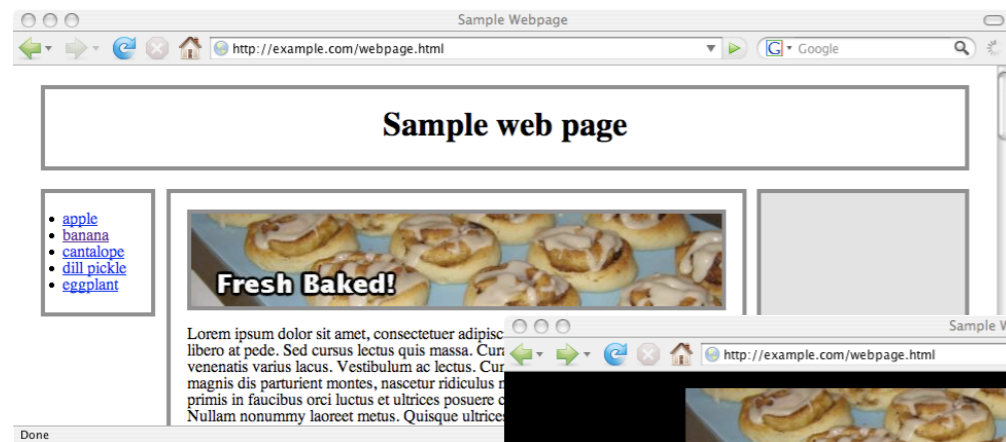
=



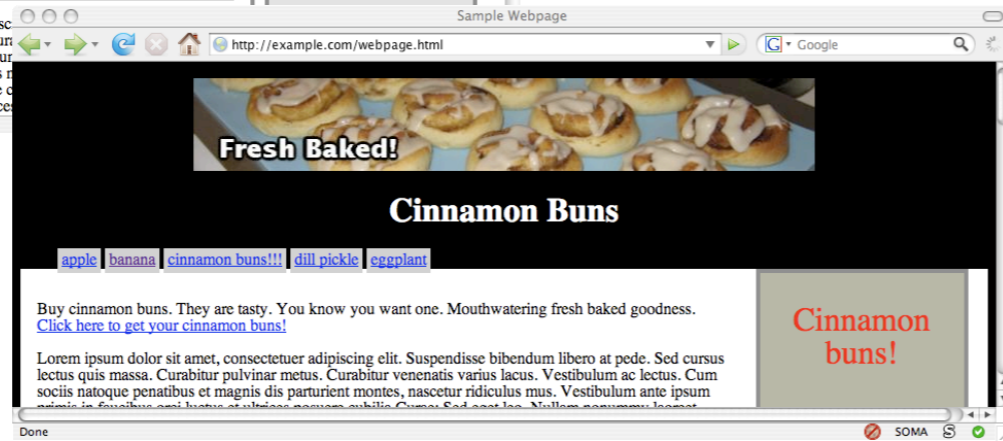
- “Static” includes go exactly where expected
- But JavaScript includes are less predictable..



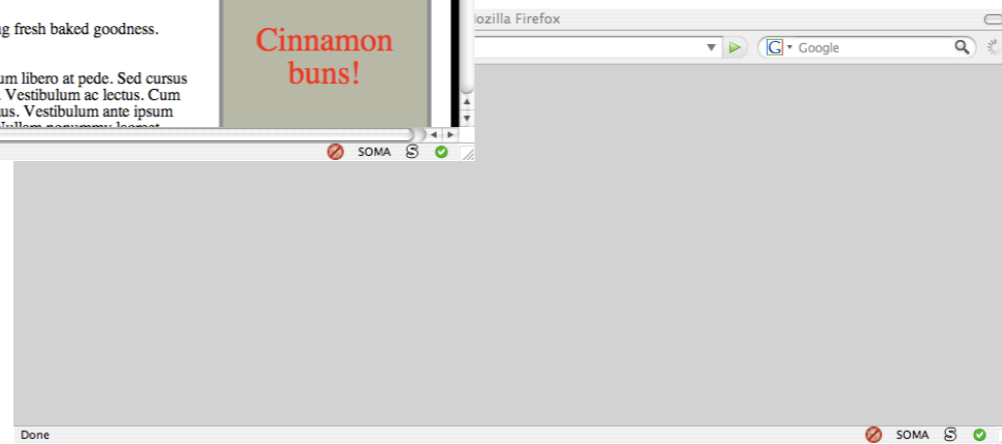
+ JavaScript =



(a)



(b)



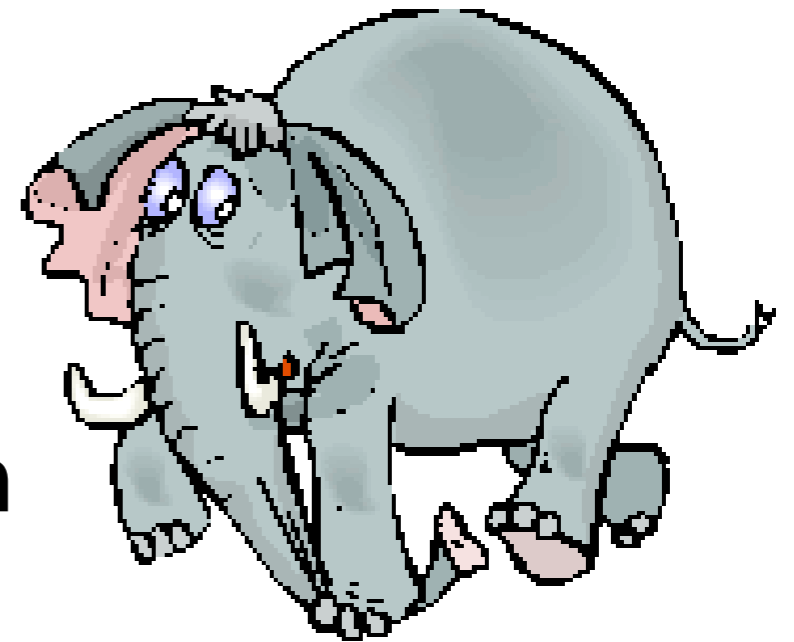
(c)

etc.

# JavaScript Security

---

- Sandbox
  - isolates programs:
    - from each other
    - from the operating system
- Same Origin Policy
  - allows communication on the same site
  - forbids external manipulation



# Same Origin Policy

---

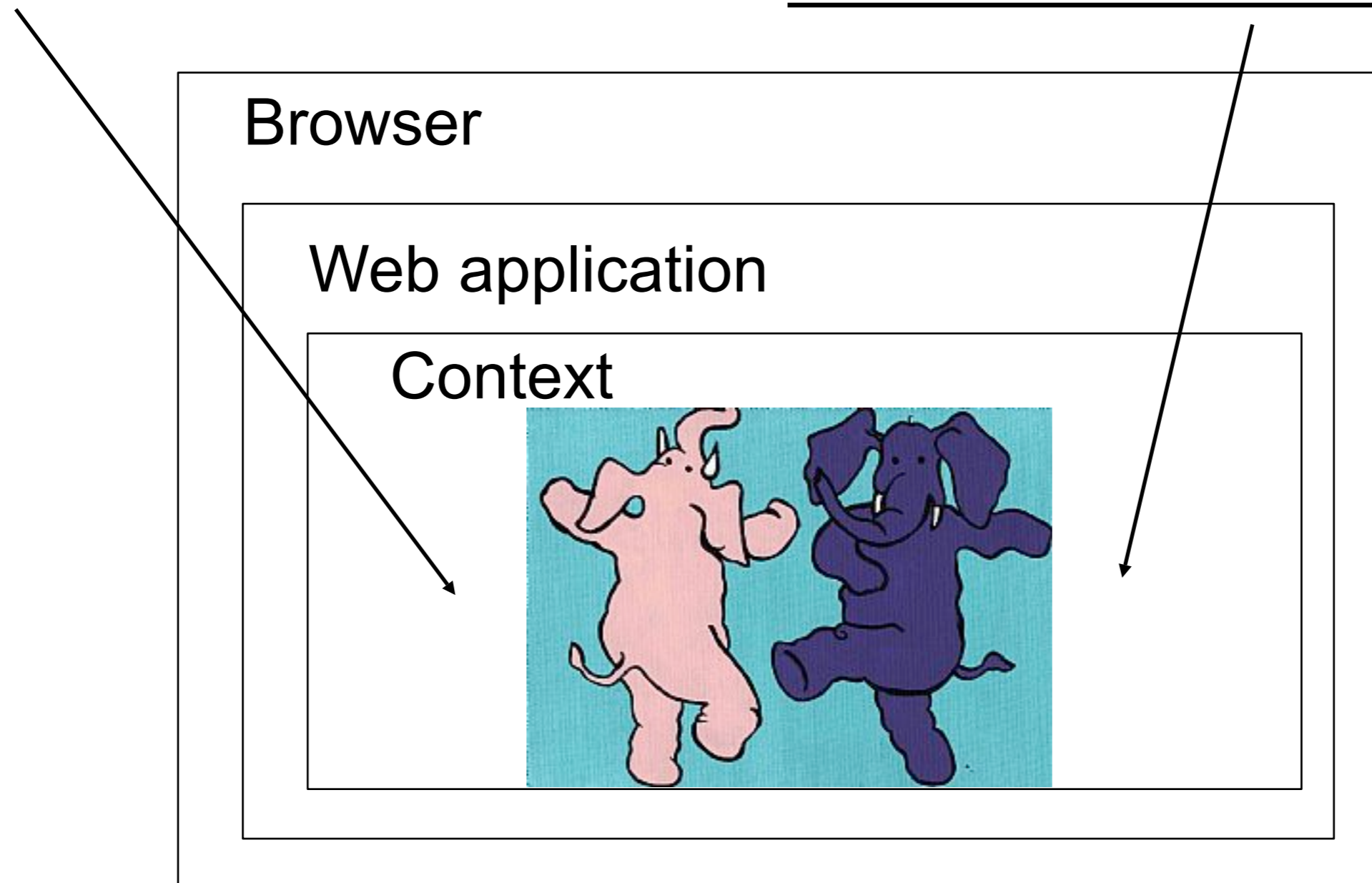
- All included code is granted the origin of the included page
- regardless of where code came from
- All included code has the same rights within a given page's sandbox
- This allows inclusion of libraries, etc.



# Same Origin Policy

[www.content-A.com/code-A.js](http://www.content-A.com/code-A.js)

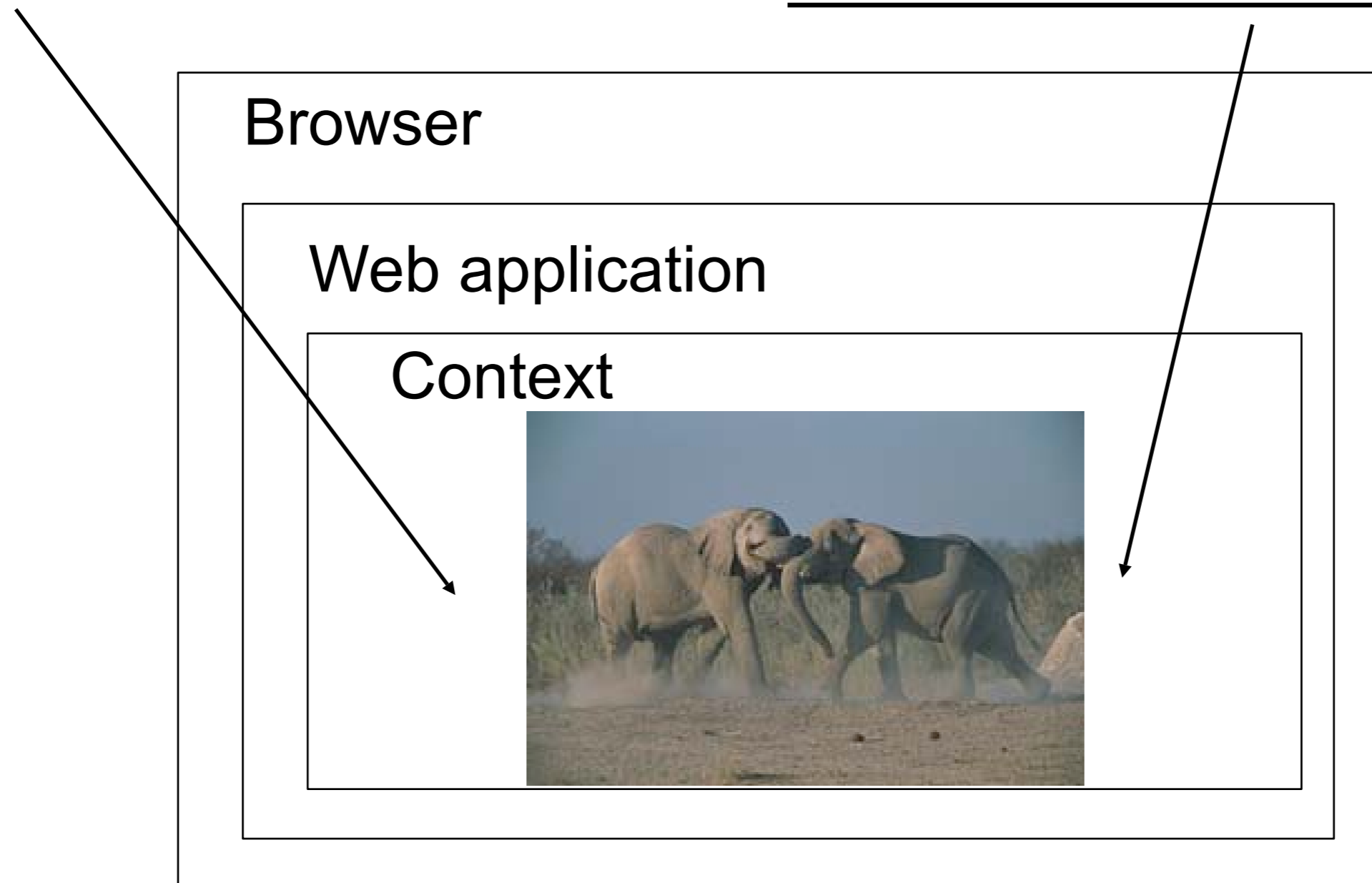
[www.content-B.com/code-B.js](http://www.content-B.com/code-B.js)



# Same Origin Policy

[www.content-A.com/code-A.js](http://www.content-A.com/code-A.js)

[www.content-B.com/code-B.js](http://www.content-B.com/code-B.js)



# Adversaries



- Focus of web security on external attackers
- But.. content providers are also dangerous:
  - Already have access to the page
    - including other providers' content!
  - May be tricked into bad behaviour
  - Have motivation to be intentionally destructive to each other

# Observation

---

- Can monitor the competition
  - Every time the page is loaded
  - With the user's settings
- See what information is being displayed to the user and when
- Can send this information back to parent site using http



# Content Manipulation



- Can alter competitor's content
  - delete entire content
  - alter/replace content
    - eg: replace ads with less appealing ones, reducing click-through rates
- Alter page content to encourage inappropriate ads, misclassification

# Server Manipulation

---

- Can send fake messages back to competitor's server
- Can pretend to be competitor's code
- send false information about the page
- perform click-fraud



# Requirements

---

1. Ease of use for all web page creators
2. Clear adoption path
3. Isolation between content providers
4. Flexibility for future innovation





# *Ease of Use*

- Complex pages are being made by “cut and paste”
- Pages are not always created by experts
  - Can't assume experts will be available to secure them
  - Users may not care about security
  - But content providers may want them to!
- Need to make tools/methods suitable to users who may not understand security

# *Clear Adoption Path*

---

- Deployment is challenging:
  - web is distributed
  - heterogeneous (software, protocols, organizations)
  - backwards compatibility
- Need to take these issues into account

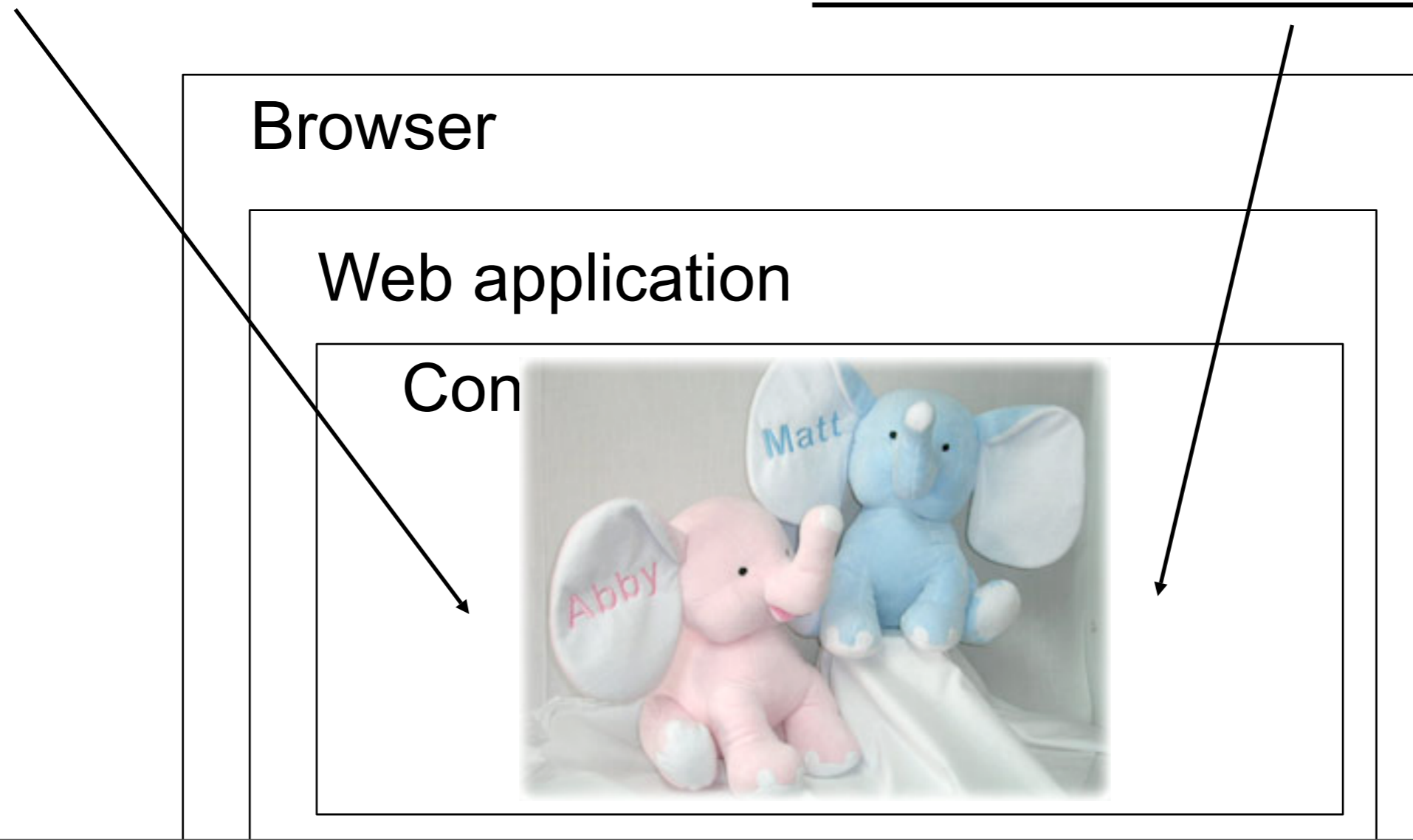


# Isolation

- Current view:

[www.content-A.com/code-A.js](http://www.content-A.com/code-A.js)

[www.content-B.com/code-B.js](http://www.content-B.com/code-B.js)

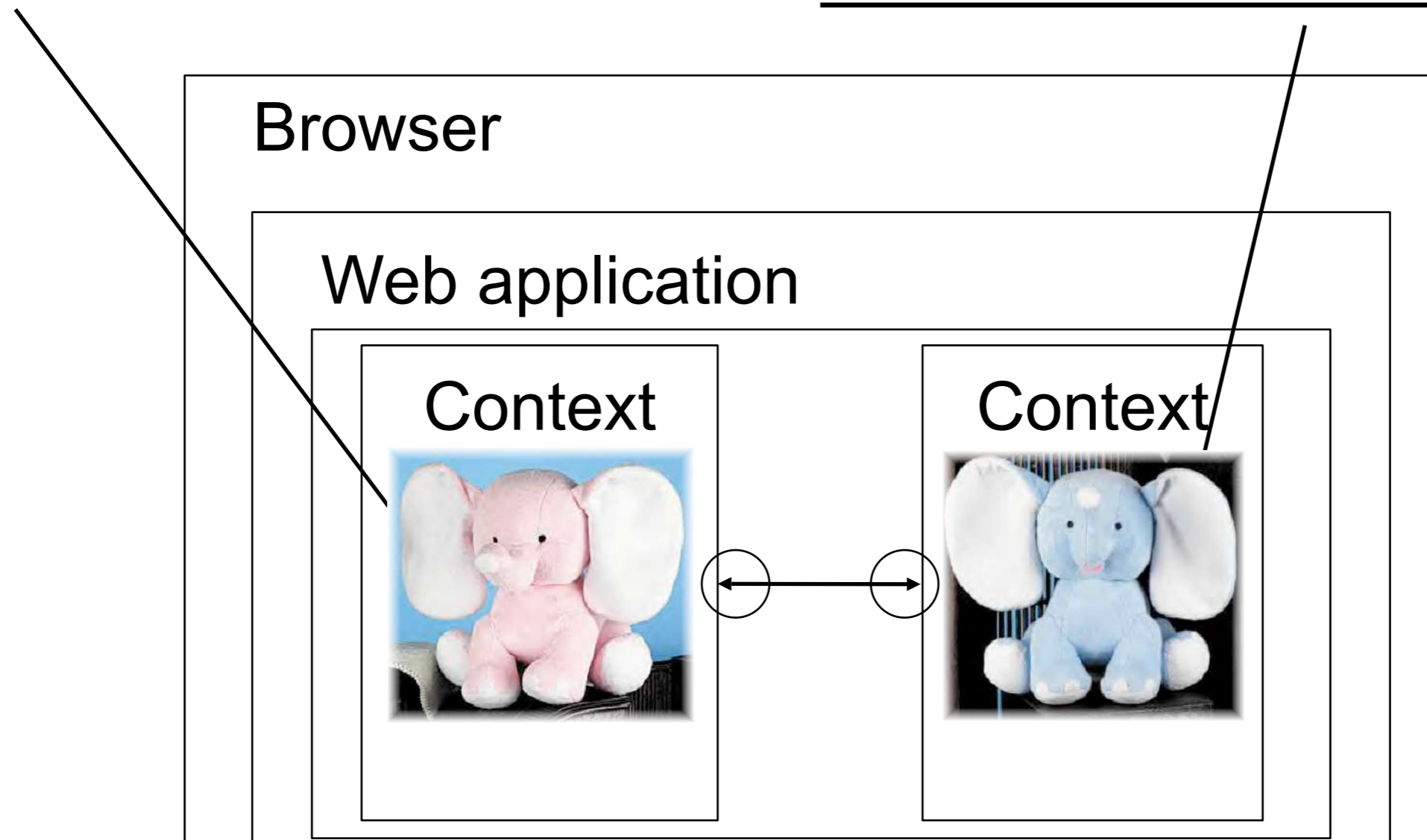


# Isolation

- Desired view:

[www.content-A.com/code-A.js](http://www.content-A.com/code-A.js)

[www.content-B.com/code-B.js](http://www.content-B.com/code-B.js)

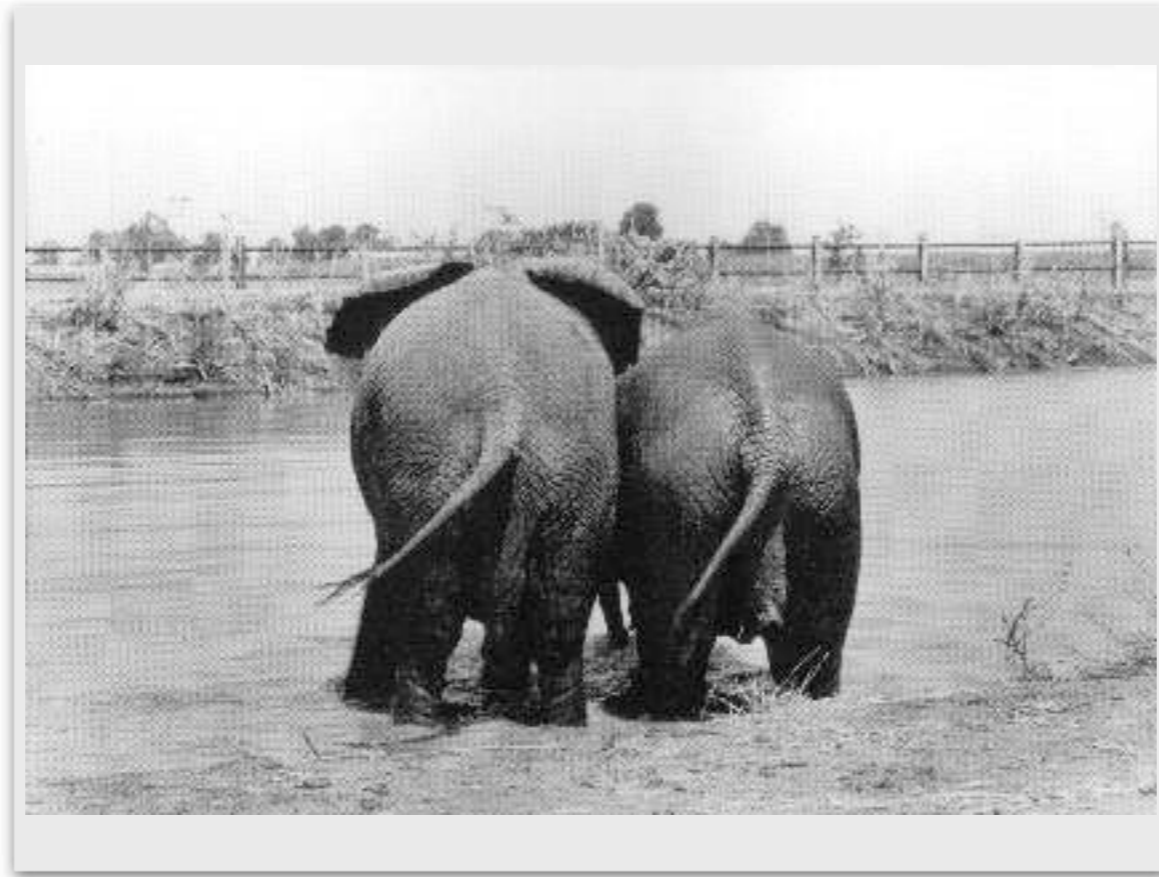




# Flexibility



- Current model of sandbox & same origin
- May seem permissive
- Allowed innovation unheard of when the web was created
- Want to block known attacks
- Want to allow for future innovations

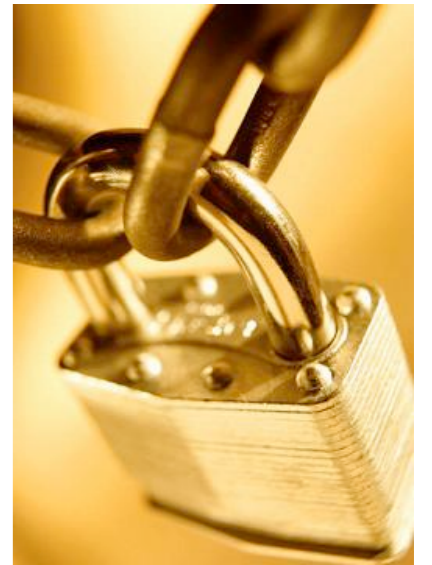


*Related work*

# Web Security Issues

---

- Web Security Issues:
  - Drive by downloads, Cross Site Scripting, Cross Site Request Forgery
- All made worse by JavaScript
- Recommendation: Turn JavaScript Off
  - But web can be unusable without it
  - Not a realistic solution for most people



# JavaScript Language

---

- JavaScript
  - full closures
  - code and data separation
  - ... already built-in!
- built-in abilities may be very useful in any solution
  - currently underused/unknown



# Content Scanning



- Possible solution:
  - Scanning web pages for dangerous input
- Problems:
  - Can't always restrict external sources
  - Advertisers and “trusted” sources may be whitelisted

# Web Mashups

---

- A mashup is a web application that combines information from different sources.
- Security research is focused on more complex sites created intentionally to be “mashups”
- But what about cut-and-paste sites?
  - Many of the same security issues apply



# Web Mashups (2)

---



- Current solutions:
  - Very good for isolation
  - May require changes to web application architecture
  - Usability bad for many who create pages
  - May overly restrict advertisers who want full-page info
  - Communication issues between content providers unclear

# Discussion

---

- Web content is “outsourced”
- What happens if businesses go bad?
- Few restrictions on behaviour
  - Some social, legal deterrents
  - But in a global Internet...





# *Solutions?*

---

- Web mashups an important step forwards
- Requirements for mashups and content providers are different
  - channels of info vs full/partial page access
  - since advertisers fund much of the web, need to examine their needs
- Ways to avoid providing full access to the DOM?
- Tools for non-programmers?



# Conclusions

---

- Code from multiple content providers = a security vulnerability
- allows observation, content and server manipulation
- May place content providers at risk from each other
- Few methods for regulating interactions within the sandbox
- Assumptions about external attackers, expert-level page creators hinder progress

# Conclusions

---



- Need to consider some very common cases, not just special mashup applications
- Need to take into account:
  - Usability for a wide range of web page creators
  - Clear adoption path
  - Separation between content providers
  - Current use cases and future innovation



Questions?