1. Begin Slide – Into, “Ask what’s the best way to improve your websites performance”?

2. Answer – Reducing HTTP Requests

3. Tell what HTTP is, go through the 5 steps on the slide.

4. HTTP Requests don’t just take time, they’re also synchronous.

5. Are they that big of deal? – 80% of time is spent on front-end downloading all the files that make up the webpage. Reduce the number of components/requests and things will get much faster.

6. Reducing Image Requests

7. 5 ways to reduce.

8. Css Properties

9. New properties added all the time. Check compatibility tables before use. Don’t discredit these properties if they’re not supported in browsers, look into modernizr and use the design philosophy “Progressive Enhancement”.

10. Does anyone remember building rounded corner images? Mention rounded corner polyfill for older versions of IE: HTML Components File.

11. That’s right, we no longer need to build gradients in Photoshop, let the browser handle it for us.

12. Here’s the code for gradients.

13. And this is what the code can look like.

**DEMO TOOL FOR GRADIENTS**

14. Next property is a cheat, it’s two properties in one. Shadows.

15. Shadows can be around a container, or inside with the inset statement. Multiple shadows can be applied, and they work with pseudo elements! Text shadow was nearly impossible before this property.

16. Demo Shadows

17. Bleeding edge property for CSS Reflections! Reflections are LIVE and they work with all the visual elements. Webkit only, Firefox has a workaround but doesn’t use this property.

18. Properties Summary: Always use them if your browser supports them, vendor prefixes may be needed for the bleeding edge. Consider progressive enhancement and modernizr.

19. next image replacement technique: CSS Sprites

20. CSS Sprite is essentially a single image container for separate images. Use in computing everywhere.

21. Walk through Benefits and Drawbacks.

22. Demo coding, two methods with either background position or absolutely positioned elements using clip.

23. Summary: ONE HTTP Request loads all images in your sprite. Any image used for layout purposes (that isn’t replaced by using CSS properties should be in a sprite).

**DO SPRITE DEMO**

24. Next Topic: Image Maps

25. Oldie but golden for what it’s meant to do.

26. Do Benefits and Drawbacks.

27. Example Code

28. Map Example, show them areas.

29. Summary, if slicing up lots of images, look to an image map. Lots of polygon areas could be done better with <canvas> or <svg>

30. Technical Piece Data URI

31. Talk about Data URI, read slide.

32. Suppor, read slide.

33. Benefits and drawbacks.

34. Show encoded image, ask “anyone have any idea what this will look like”?

35. Demo – UpFront logo

36. IE6/7 Workaround MHTML

37. Example MHTML Code

38. Data URI Summary

39. Final image replacement technique Icon Fonts

40. Icon fonts combine iconic images with typography.

41. Icon fonts are supported in just about everything.

42. Benefits and drawbacks of icon fonts

43. Talk about code and demo

44. Icon Fonts summary: perfect for what they do… all icons should be in a font! Any image that’s not, shouldn’t.

**DO ICON FONTS DEMO**

45. We can’t get rid of all HTTP requests, so maybe we can improve them.

46. Caching

47. Introduce Caching

48. Far Future Expiry Header code.

49. PHP Trick to leverage expiry headers and browser caching.

50. Cache Summary

51. GZip

52. Gzip Summary

53. Gzip File Types

54. Gzip Code

55. Gzip Summary

56. Finally, Introduce Content Delivery Network

57. CDN Summary

58. CDN Examples

59. CDN Summary