1. [Introduction] Hey there, I'm Billy White and I'm a developer and product manager at Sitemason. Today I'm going to talk a little bit about our experiences reinventing our product's interface.

First some background: Sitemason is a hosted content management platform built for agencies and designers to create websites and online applications.

Sitemason the product was first launched way back in 2001 and it's been gradually built upon ever since. Today we have this massive really amazing platform. However, our interface never really caught up to our feature set. 12 years on, there are still original UI elements that feel exactly like they're 12 years old.

So what if we could start from scratch and build our dream interface, that's totally responsive, fast as hell, and works on any device both click and touch...

2. [Brewers Celebrate] So about this time last year, Nyger Morgan knocked in the walk-off run that propelled the Milwaukee Brewers to the National League Championship Series and I had to go. How does this relate? Well, it's a long drive from Nashville to Milwaukee, and that's a lot of time to think. On the way up to Wisconsin, I got this seed of an idea that launched this whole process.

I was thinking, Sitemason, just like most services, ultimately boils down to lists and actions. A site is a list of pages. A blog is a list of articles.|| Adding a page is an action. Reordering is an action. So I got my notebook out and started listing all the pieces in Sitemason and thought "how can I put all of these pieces into a single common interface? **[Notebook Photo]** I came up with this idea to use a series of content panels that hierarchically slide over their parent.

These are my initial sketches from the car trip. I know you can’t see much, but don’t worry, I’ll explain everything in a bit.

I came back to Nashville and showed the guys, and we decided "alright, we have to build this." We really thought we'd invented this PERFECT interface… Then a couple months later Twitter came out with their iPad app and that interface essentially works the same way. And since then people we've shown have said "oh yeah, that's similar to fill_in_the_blank." So maybe we won't get any patents for inventing this thing…. but MOST importantly this interface fits OUR product.

3. Great, now we have this loose concept to go from; something to build on. But it's not worth a nickel unless you can build it. Since this was such a monumental task, before we went any further we had to define some things. **[Mission & Goals]** Mainly our goals and process. First, we kept our mission simple. Build one interface that handles

all our many assets that's usable on any device. Then make it fully responsive so it's progressively enhanced on various browsers and screens.

From here on, everything we do after this should relate back to here. The Mission.

[Process] Next we wanted to define our process. - And I promise this isn't a bulleted power point presentation. I think there's only one more slide you have to read. - Anyway, to any project manager this list isn't unique. But I can't emphasize enough the need to follow it from the beginning. You can't stay focused on these all at once, at least not with a team our size, so having these primary segments defined is extremely important.

So I'm going to briefly go over each of these segments and then show off some development stuff.

4. [Inventory] Inventory. If there's a low point in this presentation, it's this slide. There is nothing more arduous and boring (to me anyway) than taking inventory. But man, it is so important. You never really understand 12 years of software development until you have to inventory all the pieces. I have pages and pages of this crap. I spent the first couple months taking inventory and strategizing architecture alone. We have over 1200 individual pieces that need to be considered in our platform. And If you don't have all those pieces accounted for, there's going to be gotchas later in the process.

5. [Architecture] I'm sure architecture means something different to everyone here. For us, since our backend is largely staying unchanged for now, I took it to mean the architecture of the interface alone. So we took all that crap, all 1200+ pieces, and put everything in buckets and set our architecture around those buckets in a simple table. After all, if it's not simple, than the interface isn't doing its job.

6. [UX] Going into the User Experience stuff, we have to define our key concepts. And this is the last time I'll make you read.
- Content is presented in layered PANELS - a site list, a page, an article, etc.
- Anything that affects the list is an ACTION - add, delete, reorder, etc.
- Any supporting content to the panel is a TAB - iles to a site, or tags to an article, etc.
Ok, you guys got it? Alright, lets move onto the visuals.

7. [Wireframes] Ok. I set out wire framing the layouts for our primary views with Balsamiq, which is some really nice wire framing software. I don't expect you to read

any of these details in the margins, but if you want to have a look, all of my slides and notes are online at sitemason.com/barcamp.

[Wireframe Site] The first and most important layout is the site view. You can see we have the primary list of pages in a site.
- To the right here, we have those Tabs of supporting content to the site as a whole.
- Up here, we have our Actions bar where you can add, remove, and search, etc. This is uniform placement whether you're in a site or page view.
- Up in the left corner you have a nice simple way to switch sites if you're managing more than one.
- And then in the bottom right we have our always-available Help tab which I'll talk about in a bit.

[Wireframe Page] So we click on one of these pages in the site list and a Page panel slides in and overlays the site panel. Now you can work from the Page and still access the site list in the margin. And you can of course see the Actions bar in the same spot, and Tabs that support the page content.

[Wireframe Item] Next, from the Page list, you see the item content. This might be an article in a news feed or an event in a calendar, etc. Again with the Tabs of supporting content for the item.

[Wireframe Action - Add] Next, we have an example of an action in progress. You click Add from the site panel, and you get options for the type of page you wish to add and go through that workflow.

[Wireframe Shortcut Gear] Here we introduce a nice supporting feature. Since Sitemason is built to support large sites, completing common tasks quickly is really important. We've added a little Shortcut Gear per list item so you can quickly change basic settings like Title, Path, Layout or whatever. More importantly, access to the shortcut gear, and entire interface will be keyboard accessible with shortcut keys.

[Wireframe Help] And finally, one of the features we're really excited about, is this persistent help menu. Normally it's sitting in the bottom right corner collapsed, but when selected, it extends out and shares with you the description of your last click, as well as makes available all of our support resources. It's a way to learn-as-you-go for first time users, or for a quick reminder for experienced users.

8. Ok, so we have all this stuff wire framed, the next step of course is design. Now, I'm a developer and these wire frames are probably the prettiest thing I've ever made in my life, and that's only because I had Balsamiq. I am the last person you want designing anything. That's what our partners are for. **[Design - Kevin]** So we worked with this guy. Kevin Kennedy is a fantastic designer with our partner Signal Hill out of North

Carolina. He's been invaluable in this whole process, and not just design. Anybody looking for a good creative partner, definitely give these guys a call.

[Designs ALL] These are the designs that Kevin came up with based on our wire frames. We've got our main site panel, a page, an item, an example of adding a page - giving it a title, deleting some pages, advanced search, our shortcut gear, our expanded help menu, and finally a couple notifications.

9. [Front End Dev Considerations] Alright, we have the designs, everyone is super excited… now we have to build the damn thing. Before we got started we wanted to list the goals of the front end. How we want the interface to *feel*.

First and harkening back to our mission, we need this thing to be responsive. Since the interface is in development and liable to break at any time, I have a video showing off our responsive breakpoints to give a rough idea of how this thing might work on devices large and small. **[Responsive Video]**
- Alright, we're looking at some of the basic interface functions.
- Notice how the hover is never required but enhances the desktop experience.
- We've got a large list of pages here, and everything is hierarchical for large sites.
- And finally our breakpoints

It also needs to be fast. People are busy, so they can't be waiting on us. As you saw previously, the panels are pretty quick. The Actions need to be as well. Here's another quick video showing off those Actions. **[Actions Video]**
- Here we add a page, pick the page type, give it a Title and a Path, select where to put it in the list, and create
- Next we reorder the list. Notice everything is drag and drop
- Open a page, add some content, save.
- Here we duplicate a page
- And finally delete that duplicate

One byproduct of the responsive design movement is that you can no longer rely on fixed pixel anything really. So we set out to make our interface without relying on any images. We decided to use the icons that Kevin had designed, and turn them into a Font Icon Set.

Font icons are literally fonts. They're icon vectors mapped to character values. If anyone remembers Webdings, those are font icons. A great example of the benefits of Font Icons is at Css-tricks.com [**exit and minimize Keynote. Pull up Chrome & Css-tricks.com**]. As you can see you can apply all sorts of changes to the icon without having to build a bunch of image sets. And here's our font icons that Kevin created for us. You can copy these icons, and paste them somewhere to see their character mappings.

10. [Development]

Here's where we are with the interface [**pull up SM6 tab**]. This is very much unstable pre-alpha software, so we can't do a whole lot in it, but you can see the panels slide, and content lists load, and everything's really fast and slick. This is real content from the Sitemason website.

We wanted to start out with pristine semantic html, and we've done that. [**Inspect Element**]. Inspecting Element on the interface, and just check out how clean this markup is. Also notice how few resources we're loading for such an insanely complex interface. Looking in the <head>, we only have two stylesheets. One for the Icon Fonts, and the other to style the interface.

The interface only relies on two outside libraries. We are using jQuery, the javascript framework, for some low level browser-unification and server connection stuff. And Modernizr which does browser feature detection. If you haven't used Modernizr before, all these classes on the <html> tag here are browser features detected by Modernizr. We currently only use it to detect if a browser has Local Storage available.

[pause]

For our interface, the biggest concept to understand, and this may take a second to sink in, is that the entire interface is generated by javascript. Let me repeat that. The ENTIRE interface is generated by javascript. There are no HTML or PHP files.

So all the markup you're looking at here, it's all created from javascript config files. After it's loaded, the only calls to the server are to retrieve data as JSON objects. So the interface and the data are literally all javascript.

The retrieval, delivery, and display of EVERYTHING, is all done with javascript. This is truly a platform independent interface because the platform IS the browser.

So for an example of this, I took a snippet from sitemason.js **[Show in Inspector]** which draws the entire interface, to show you how the Action Bar is generated. [**Pull up Coda with actionBar.js**] As you can see, its just a JSON object. Up here we define our Action Buttons which is just a list of buttons with their behavior. So we've got an Add button, which is mapped to the "A" fontIcon. It has some values like title and submission URL. Then cascading beneath that are options for the workflow to add a page. The option for which type of page, give the new page a title and path, and finally the submit button.

Then we do this for each of the action buttons like Delete, Reorder, and Duplicate [**highlight each**].

Do it a few dozen more times, and we have our entire interface, built exclusively from javascript.

Earlier I mentioned Local Storage. We've really come to realize that all of this JSON data we're pulling to populate the interface could be stored in Local Storage. **[Inspect Element - Local Storage]** If you aren't familiar with web storage, it's an HTML5 feature similar to cookies but with much much more storage. So you can put huge data objects in Local Storage and treat it almost as a simple local database in the browser.

The more we've worked with it, the more we realized, "hey, why don't we put all of this data into Local Storage?" Now, we've got it down to the backend doing very little more than long term storage and retrieval. All of the display and most of the functionality, is all happening in the browser.

So Sitemason uses Local Storage quite liberally. Having all the data locally on your device makes sorting on hundreds of items become trivially fast. You can see that here on the Sitemason blog, which has about 100 items **[Sort List]**. With search [**search for 'part' then 'partners'**], you can see how quick it is updating as you type. We actually had to put a pause between key strokes because it was too fast! Pretty neat stuff.

END:
Alright, well that’s where we are with the new interface. And that about wraps up my session. We have a few minutes for questions, if anybody has them.