For anyone who enjoys (or has a knack for) planning, [**organizing a hackathon**](http://techcrunch.com/2012/03/31/hackathon-planning/) is not terribly difficult: it’s a matter of understanding your goals, assessing needs, and figuring out how to bridge the two. Naturally, this is much easier said than done.

The most important part of a hackathon, by far, are the demos. I mean, they’re what make the event worth attending in the first place. Sponsoring companies wouldn’t offer money to anything that didn’t provide exposure. Developers wouldn’t forsake sleep if they couldn’t show an eager audience the hacks they built overnight.

**Format**

Almost every sponsored hackathon will have two sets of presentations: those that kick off the event (for sponsors to introduce developers to their APIs) and those that close it (for hackers to show what they’ve done). Since company demos are the most visible component of the hackathon, ensuring that this portion runs smoothly is key to the event’s success. **The caveat: extensive preparation is useful to a certain degree.** You can – and absolutely should – do a number of related tasks in advance, but a significant portion of hacker demos require efficient, on-the-fly work that takes place within a short amount of time.

Photo Hack Day 2 was organized almost entirely through three tools: email, spreadsheets, and[**HackerLeague**](http://hackerleague.org/). How you choose to keep track of everything is up to you, but I’d strongly suggest the following for each set of demos.

**Part I – API demos:**

* *Keep a company database handy.*Any correspondence with a particular company should be labeled and filed away. It wouldn’t hurt to note how up-to-date companies are with their essential obligations, such as a cross-promotional blog post, whether or not they supplied a logo for the website, or when they paid any sponsorship dues. A CRM will help streamline the inbound and day-of workflow, but keeping everything in a consolidated document in a single browser window saves a significant amount of time compared to digging through archives. I’ve included a [**sample spreadsheet**](https://docs.google.com/spreadsheet/ccc?key=0AruC_fn-fSY0dHNCSkFPRkZidUd1MlN1alZHVzhKUlE)where I organized the relevant information, and left a few examples of notes that I didn’t want to forget. (Note: the template is the same as what I used for both Photo Hack Days.)
* *Encourage presenters to use layman’s terms for sake of efficiency.* If hackers are curious, they will approach a company developer and ask questions after a presentation is over. Being overly technical hampers presentation speed.
* *Distribute guidelines on good presentations ahead of time.*As a rule of thumb, attendees and developers are generally fond of the following: Cool APIs, [**funny**](http://vimeo.com/40255931) presentations and no PowerPoint.
* *Establish time limits and follow the schedule.* Higher-level sponsors generally demo longer, and with more materials. Giving your headline company five minutes to speak in front of a captive audience with a laptop and projector is ample time; the next tier of sponsorship can have three minutes with a laptop and projector. Non-sponsors are allowed a one-minute pitch with a microphone, and can host non-mandatory demos and workshops during hacking hours instead.

**Part II – hack demos:**

* *…Clearly establish time limits* (yes, it’s important enough to mention twice.) – Unlike sponsors, each team of developers gets an equal amount of time to present. Two minutes is standard and a well-planned presentation can cover the high-level ins and outs of a hack in that much time.
* *Ban PowerPoint if you have to* – This is also worth mentioning twice, because too many teams continue to try to use it. It may even be worth an explicit ban. It’s a waste of time and they can’t effectively showcase an app or service’s functionality.
* *If companies are providing their own sponsored prizes, supply a list of hacks (and, if possible, the demo order) that can be filtered by API usage -* This enables them to know who to look out for; shareable spreadsheets on Google Docs are perfect for this.
* *When s#\*t breaks — and trust me, it will — have multiple spares on hand -* During Photo Hack Day 2’s Demo Day, we destroyed General Assembly’s two HDMI and VGA cables (they were taped to the floor underneath the audience chairs, so when people moved around, the cables were yanked from the wall and damaged). So, keep two or three spares of everything to keep the party going: electricity, Wi-Fi, presentation tools, dongles, cables, power strips, extension cords and even a projector! (Pro Tip: keep your receipts!)
* *Prepare any relevant material for the judging panel -* Provide all judging criteria ahead of time and make this publicly available for attendees and hackers alike. Provide plenty of writing materials and judging templates (usually a list of hacks; this falls under the category of things you’ll need to create on the fly). Encourage judges to ask questions and push back on developer ideas – for instance, “Why did you do it like *x* when you could have done it like *y*?” If hackers can consolidate important development points in ten seconds or less, that will definitely win over technical judges.

**To-do on the fly**

Remember the aforementioned caveat? This is it, and it’s the trickiest, most stressful portion of the event.

* *Demo order – C*onsolidate signups into one place. Categorizing demos based on requisite equipment (i.e. live websites, locally-hosted applications, mobile apps), will prevent you from having to scramble to find the appropriate cables / dongles every time a new person comes on deck. Test your platforms thoroughly; you’ll need to be very familiar with the sign-up procedure that your hackers will be relying upon.
* *If you’re using social media to post updates, create incentives for people to follow relevant feeds -* During Photo Hack Day 2, we posted most of the updates via Twitter with the #PHD2 hashtag. Everything seemed fine until the demo sign-up process was announced, both over the microphone and via Twitter — Not everyone could hear us, nor were they checking Twitter.

By creating incentives (i.e. sponsor giveaways to those who follow the feed, tweet about the event, or retweet any official posts), you gain a stronger following while making sure that hackers stay in the loop. This translates to a less chaotic process of organizing demos.

* *Voting process* – Whether you choose a web or mobile service, it helps to have SMS voting enabled. I say this because (1) I have yet to find a platform that I love to use on mobile browsers, and (2) most Demo Day attendees are not going to bring a laptop with them. Be sure to test it ahead of time to avoid hiccups.
* *Prize ceremony -* If there are only official event prizes, the process should be easy enough: emcee as necessary, but allow judges to briefly talk the audience through some of their thoughts before delivering the verdict. For company prizes, be sure representatives know when to be onstage. Keep timeliness and brevity in mind. Feel free to relax now.

**Here’s what I still haven’t figured out**

*Should there be a cap to the number of demos?*

This has some very obvious pitfalls, but after four hours of demos, the judges and audience at Photo Hack Day 2 were equally exhausted. This translated to a rushed prize ceremony: it was getting dark, everyone had been at General Assembly since lunch, and hackers were beginning to shut down after a night without sleep.

If judges and attendees are only interested in the high quality hacks, should there be some sort of screening process? The tradeoff for a large turnout is a long Demo Day, and you’d hate to end the event on a poor note (read: tired, grumpy hackers and attendees.)

*What alternatives exist to the monotony of demos? Is it worth dividing hack submissions into categories?*

After the hackathon (and a much-needed day off), our team broke down the areas where we thought there was room for improvement – and there were plenty. Though the demo tech mishaps were the most glaring, the actual process of demoing needed work.

The consensus was to encourage (and more importantly, reward) creative and entertaining presentations with prizes; they provided a much-needed relief to the spell of monotonous ones. Good demos – especially those that are funny – make the entire afternoon worthwhile. In terms of condensing the block of time required for participants to demo, the jury is still out.

**A final note**

This is the summary of everything I’ve seen and experienced from the two hackathons I’ve organized, and the three cumulative months I’ve spent planning. There are certainly alternatives to all of these “methods” and a better way of approaching these things is to consider them variables. Trust that you understand the nature of your hackathon well enough to know what will work and what won’t.

Yes, demos are time consuming, and yes, they are stressful, but I absolutely wouldn’t mind doing it all over again. For the developers who appreciate cool APIs, for anyone who loves observing the creative process, and for the many people out there concerned with building community: organizing these type of events is so gosh darn rewarding. Awesome people should meet awesome people, and awesome products beget even more awesome products.

This, in a nutshell, is why we put on hackathons in the first place.