Example
Tech Documentation

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Introduction to Developer Evangelism

This Developer Relations Evangelist Guide is a combined collection of experience from Christian Heilmann, Senior Program Manager Developer Experience and Evangelism at Microsoft (ex Mozilla and Yahoo!) and myself (ex Opera Software ASA and Google/Motorola Mobility). Christian is well known for his dedicated excellence and practice in the world of Developer Relations and Evangelism, while I contributed technical developer and product documentation to the developer relations teams where I worked. *Frank M. Palinkas, Senior Technical Writer.*

Purpose

This guide is intended for motivated individuals who aspire to become Developer Evangelists and understand their company's Developer Relations process. It will attempt to get you on your way to be a great developer evangelist for your company and its products. Your approach will need tweaking for different markets and audiences, but the main principles are the same for everybody, anywhere. Developer evangelism is a relatively young field of work and the first hurdle you will encounter is people asking what a developer evangelist is and why a company would need this role.

What is developer evangelism?

A developer evangelist is a spokesperson, mediator, and translator between a company, its technical staff, and external developers who are using your products, are thinking of doing so, or who are completely ignorant of them.

Who should become a developer evangelist?

As a developer evangelist it is very important that you used to be a developer. The more projects the better. Your job will be to make your company's technical offerings attractive and interesting to a large variety of developers, and you can only do that when you know their pain. A developer evangelist is a role that is a *change for developers*, not for people coming from HR, PR or marketing. Your main job is still to code, but this time code examples, training materials, and explanatory demos rather than live products.

Your developer audience

Developers make Information Technology work. Ideas start a great product and UX Design makes it work for users, but to make it work technically you need developers. Sadly enough developers do not get much credit for their work and are generally considered "deliverers" rather than "thinkers", which is simply not true. What this also means is that telling something to developers as a company or getting them excited about your products is quite a task.

The trick is to understand that to be a developer (especially an app or web developer) you need to have a certain way of seeing the world. And this way of seeing the world makes you suspect things to fail in any which way. If your message means less work for the developers out there it is a great start for you. If it means extra work on top of what is already on their plate (and developers always get maxed out) then you may have a more difficult time.

Starting with the right mindset

The main thing never to forget as a developer evangelist is the technical part. It is very easy to get into the habit of just writing one presentation after another and re-use materials but this way you will not have much impact.

If a new product comes out of your company that should get out to developers, take it and access it like an outside developer would. Integrate it following its build instructions and document "what" you have built. Then write "how" you built it and you've got half an article or presentation already finished.

Find your role and use your strengths

Not everybody can and should be an all-around evangelist. It is enough if you find your place in the whole spectrum of evangelism. Think about *what you love to do the most* and then start creating something. The most common niches of the whole job to go into are:

- Writing code tutorials
- Blogging
- Public speaking
- Training
- Social web coverage

Study the rest of this guide and see what best suits you, then start evangelizing. You have nothing to lose and will most probably be very surprised how easy things become if you concentrate on one job at a time.

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Building code examples

Contents

Overview

Code examples make your posts and articles very relevant to developers. They say more than dozens of pages about a certain product, and more importantly, they invite people to play with your products.

Solving a problem with your example

Point: "Hello world" code examples don't do anything useful. Instead, aim to give out code that solves a real problem. Writing "Hello world" code teaches people to write code, but *not how to solve issues with it*.

Good code examples should answer the "how does this technology help me solve a problem?" and **not** the "how do I use this?" question. The "how do I use this?" question can be answered by documentation. Code examples should get people excited about using the product and entice them to dive into the documentation to find out about the details.

Instead of starting by reading the documentation, check what the tool can do and look for a problem you always wanted to solve that can be solved with this technology. *Then solve it using the product and explain what you've done as the code example*.

Show a working example

One of the first things to show in a code example is a working implementation. For example, there is nothing more powerful than a way to see what the thing does by clicking a link or entering some data and sending off a form. You telling the readers that it works is one thing: the readers being able to try it out and seeing it for themselves is much more rewarding.

> **Tip:** If the code you explain is part of a larger interface, behind a firewall, or needs authentication, then you can still show how it works by *recording a screencast*.

The working examples should of course work but also be pretty and smooth. A lot of examples fail to please the eye or actually violate a lot of usability basics: don't give a shoddy first impression.

Explain the necessary environment

One of the things you want to avoid is people getting excited about your code and then not being able to make it work in their own environment. You get around this issue in several ways:

Write defensive code.
Check for dependencies before trying to access them.

Check online and HTTP status.
If your code needs to pull data from a web service, say that the user needs to be online to run it. For example I've had many a complaint that my Ajax examples don't work offline or without running a localhost. Obviously I failed to explain that Ajax is meant to work over HTTP.

List what is needed.
Say what your specs are, for example: "needs PHP5 with cURL and imageMagick" or "Firefox 35, Opera 29 or Safari 5x needed".

Provide a simple test script.
This will check for the right setup.

Provide a developer key for the demo.
If necessary, make it obvious to tell people that to implement your code they'd need their own key. Provide a link where to apply for one.

You won't be able to predict all things people can do wrong when implementing your code but these are some good ways of preventing frustration. Which brings me to a very important part of code examples: allowing for copy and paste.

Write working copy and paste code

Copy and paste is probably the most used way of learning code. You can write documentation until your fingers bleed, but the biggest use case is that developers will check a code example, copy and paste it, fiddle around with it until they get stuck *and then start reading the docs*.

Therefore it is immensely important to write very good copy and paste examples. Any bad coding habit you add in your code will be copied and become part of a live implementation.

Copy and paste examples that don't work are not only useless but also disastrous to your reputation and very frustrating for implementers. Therefore make sure that the following points are covered:

Link all resources.
Point all images, CSS, and script resources *to web locations* rather than linking them locally or relative. People will copy the code and paste it in a local HTML file and not download the dependencies.

Provide the full script upfront.
People will copy and paste chunks of a script and complain that they don't work without realizing that other parts are missing.

Validate and secure your code.
Copy and paste code needs to be excellent as in many cases it will be what people use. Make sure your code plays well with other code and doesn't cause any warnings or errors.

> **Tip:** The best way to keep code examples and code in sync is to generate the code from the source comments.

Have the code example(s) as a download

Providing the demo code for download as a zip file should be one of the first things you do in your example. You can ask readers to download, unzip it, and code along with you (this works well with screencasts and video tutorials). In any case it keeps your article in the mind of the readers as they have something on their hard drives reminding them of it.

Offering a zipped version of your demo code can be annoying as every change means you have to re-pack the demos. Luckily by using a hosted code solution like GitHub this job is done for you automatically.

Writing clean and clever code examples

Again, we cannot stress enough that **your code examples should be the cleanest and cleverest code you ever write**.

It is very tempting to show a quick and dirty solution to get people going and earn immediate kudos for creating the shortest code ever, but this is what coders do to impress each other *and not what evangelists do*.

You want to show how to write excellent solutions with the product you evangelize, and every shortcut you take will be copied and taken as an excuse not to write excellent code in live projects. This is not what we are here for.

On the contrary, demo code can advertise best development practices and show them in context rather than just as an academic exercise. Showing a JavaScript that protects scope by using the Module pattern and uses closures cleverly allows you to cross-link to these ideas and maybe get some developers to take them on as part of their coding habits.

Building code generators

A very nice touch to add to a solution are code generators that allow implementers to just add some parameters, hit a button and get the code they wanted. This is amazingly powerful.

The only danger there is, is that some people will never bother learning the real implementation tricks, but on the other hand these readers are not likely to do that anyways. In any case you'll get a lot more people to look at the product.

Example
Tech Documentation

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Creating posts and articles

Contents

Overview

Writing for the web is a specialist skill and far too often you'll find people applying rules of other media when it comes to writing online articles or blog posts.

The main thing to remember is that text on the web is not published in one place but spreads throughout the web as bookmarks, links and references. Therefore it is very important to chunk texts up into easily digestible and repeatable chunks and make your headlines and document titles work without the context of the full text.

Being simple is not stupid

Writing very simple texts is hard work. Writing stupid texts is less hard. Consider the following traits of simple texts:

- Writing in simple terms takes a lot of work and thorough understanding of the subject matter. You need to be very familiar with the topic to be able to explain it in very easy terms.
- If you explain things in as easy as possible terms you make sure that you reach the largest amount of readers.
- Simple wording allows non-native speakers to get a chance to understand what the whole thing is about and maybe spend some time with a translation tool to make it work for them.

There is however a fine line between explaining complex things in simple terms and sounding condescending. Have someone else look over your texts to avoid this issue.

Read and re-read what you’ve written (take breaks in between) and make it easier in every iteration. Comparisons with real life objects work very well to simplify complex matters.

Saying what it is

Your heading and introductory text are the most important things of a blog post. Both determine how easy it will be to find the post in the future.

Newspapers have conditioned us to write clever, witty and interesting headlines with pop references and alliterations. That is fun, but you already have bought the paper once you read them. Web headlines become links and bookmarks and need to make sense without the rest of the text. For a technical blog post state what the post is about, do not try to be too clever. Pop references are even worse as these don't translate well to other cultures.

So do you want to be creative and witty for a minute or do you want to provide valid information for several months? Blog posts should work like news items in radio:

- At the start of any post state what happened, where and how.
- Continue to explain what is coming in the post.
- Then go into details.

This will prevent any confusion and put interested people on the way to find out more. If you wonder where your blog headline needs to make an impact just think of RSS readers and personal home pages. A headline of an article or blog post and is where it counts to be informative. Does your headline both tell the reader what is going on and entices them to click it?

Size matters

Writing for online use is about keeping things short and to the point. People are busy and want the facts. So in order to write great posts, write them, read them, delete what is not needed, read again, delete more and so on. If you cannot take anything away any longer, you've reached the point of publication.

If you have a lot to cover, why not split it up into several posts? This will allow you to tease at the end of the first post that the next will be published in a few days and give you repeat visitors and readers on your blog.

Adding media

If you can, add relevant media to the post. An introductory photo invites the eye and lures the brain into reading what happened. We're lucky that these days embedding video, audio, and slides is as easy as copy+paste.

Embedding ties our information together in a nice, easy to digest bundle. It also allows visitors to skim over the post the first time and come back to take in the rest (watch the video, download the podcast) later. This also helps people who have a hard time reading but are very much capable of listening or seeing.

When embedding media make sure to also write explanatory text. Images need good alternative text and videos at least a description of what they show.

> **Tip:** Slides get much better by providing a text alternative to your notes. SlideShare automatically creates text versions of your slides for you.
> Start with your notes and then create slide decks from them. This way you always have a HTML version to link to from the deck.

Structuring your content

Structuring your content is very important. Giving readers landmarks to take in your information one chunk at a time allows them to skim the post before deciding to read it, and as people are busy this goes down really well.

You achieve this by using a hierarchical heading structure. This will also help people with assistive technology like screen readers to jump from section to section quickly. If you add IDs to the headings people can bookmark parts of the document and send links to friends that will bring them directly to where they want to be.

Structuring a text also means using short sentences. It means paragraphs dealing with one thing at a time. It means using lists to explain step-by-step processes or give an overview of what is available. For large documents it also means providing a table of contents which allows people to jump directly to where they need to be.

Time-stamping your content

If you eat food past the "best before" date you get sick. If you don't time-stamp your publications they will be considered great forever – even if they are harmful by the technical standards of the future. They'll be quoted, sometimes badly, and re-iterated over and over again and taken as gospel.Our technical environment moves at breakneck speed. What was "best practice" half a year ago might very well be "considered harmful" now. So let's make sure that readers know when a certain document was written before following its advice even now.

Citing to prove

Another very important point in this section is to cite other sources and link to content you have built upon. By citing other sources (and reading them of course) you validate your thoughts and facts. Readers don’t have to trust you blindly: they can make up their mind by comparison.

Preemptive writing

One of the main things to remember is that as a developer evangelist you are the missing link between tech people and the outside world and tech people and your own company. This gives you a task that no other company department doing outreach can fulfill This is called preemptive writing.

What I mean by that is that when you talk about products you should not sell them. Instead your job is to make them interesting to developers out there and let them be the sales people for you. The main task is to anticipate feedback you'd get if you did a plain old sales-pitch and answer that feedback **before** it happens: preemptive writing. In most cases this means one thing:

Don't try to sugar-coat or hide the negative.
Instead explain why it happens, what to do when it happens, and how to report to the people whose job it is to fix the issue.

As developers we are a cynical bunch, battle-hardened by years of broken promises. An overly excited and positive description of any product is not going to make us think "I have to get me some of that" but is more likely to spark an "OK, where's the catch?".

This makes developers appear not the most social bunch, and it can be incredibly frustrating for a Marketing or PR person to do a classic sales pitch just to be faced with a barrage of negative feedback. Your job is to predict part of that feedback and disarm the situation by acknowledging it in your writing. This can be a hard sell to your PR and marketing departments but you can explain it to them like this: by being open about the good and the bad you avoid a lot of confused or bad feedback that will be taken out of context by the press and cause the PR department a lot of unneeded repair work.

Negative feedback in a lot of cases falls into a few categories, all of which are commonly showing the writer in the light of an "internet troll". Here are a few categories of negative feedback and tips to preemptively avoid them:

This doesn't work at all for me: it is broken.
Make sure to very obviously define what environment is needed to run the product or demo you talk about.

This is the same as product x by company y.
Many times this is a **fanboy** of a competitor speaking. Mention that your product is very similar to others and list them. Explain in detail what the differences are and that your product is more than just a copy. By showing that you know about the other products you prove that you did do your homework and still found it worth-while to talk about it. Do not let people catch you not having done your research.

Great but I can never use that, nobody needs that.
Explain in detail *what problem the product solves* and if this is not a very common problem now but one of the future ask for feedback on how people like the product. Also tell them that this is a chance to make sure the product will be ready and great once they encounter that issue.

Why do you waste time on this when your product y is still broken?
Make sure your writing sets the scope for the article. Talk in detail about the one product. If you are lucky, other people will answer to a comment like this that it is out of place. If not, answer swiftly that it is and point at a different thread or discussion where that person can vent their unhappiness about the other product.

I like this but I don't have time to test it.
Prepare demos and code examples with simple feedback mechanisms for people to try out with a simple click. Maybe even offer a simple survey.

These are just a few examples of what can be done to preemptively avoid bad feedback using full disclosure. There are many more but I am sure you will find good ways to deal with the issue.

Establishing your technical integrity

Your integrity builds trust. Trust builds credibility with your developer audience.

As a developer evangelist you have to keep your independence. By vocation you are a specialist in the technologies of your company. However, if you are oblivious to the world outside the company and preach a corporate-branded mono-culture you will not get far. Remember: product first.

Your *independence* and your *tech integrity* are your main weapons. If you lose these defining characteristics you will no longer be effective. Developers should get excited about what you do because they **trust your judgment**, not because you work for a certain company. This means that as soon as you exclusively start talking about your *brand* the cards are stacked against you. You'll preach to a choir, leave most of an audience bored out of their minds, or get shot down in flames.

Developers are passionate about and loyal to solutions and technologies *once they are happy with using them*. The really fun thing is that every company out there wants to do what you want to achieve: make developers happy using their products. Therefore it is very important that you keep an eye on and in be contact with the competition as much as it is important to be aware of what is going on in your own company.

Respecting the competition

You cannot be a professional evangelist and bad-mouth the competition at the same time. We all are professionals and work on projects that strive to make our particular product environment a better place. Pointing out weaknesses of the competition's products is a cheap shot. To experienced people, this signifies product weakness and thus reflects badly on the brand/company being represented. The world of developer evangelism is very small. Cross one respected product evangelist and they will soon come back to you with speed.

Acknowledging when the competition is better

This is a hard pill to swallow for a lot of people (especially for marketing departments), but bear with me. If your competition has a better product than yours and people ask you which one is better, *do admit that this is the case*. Whilst this sounds like admitting defeat in reality it shows a few things you can use to your advantage:

- You come across as someone who appreciates good technology.
- You come across as someone who does not fear competition but welcomes it.
- Your competition feels that they've done an amazing job and will look closer at your products in return.
- You will learn what people love about the competitor's product and can feed that back to your own product team.

Another thing to remember is that the competition's product might be better **but for a different audience**. Remember your product's strong points and target market - don't allow a comparison of apples with oranges to take place.

Knowing the competition

This is a classic marketing, advertising or even development step: before you build or promote something look around and do a competitive analysis.

In the case of developer evangelism you need to be up to speed with what your peers are up to as you will constantly get questions about it. "How does this compare to X, the new product by Y?" is a very common first question. If you can answer that, *your tech integrity* will get quite a boost.

Building mashups using competitive products

Using the web services of your competitors is a great way to check several things:

- What do they do well, that could be done better in your company's services?
- Where do you get stuck? This is something to avoid in your own services.
- How does it compare to your own services? What are the differences? Remember, these are the questions that people will ask you, and you learn best by doing.
- How can these services be mixed with your own and how do they compliment each other?

Removing your brand mindset

A very crucial part in becoming a successful developer evangelist is to remove the brand from your thinking. Yes, you work for a certain company that builds products. The point of developer evangelism is not to get people excited about the brand or the company behind it.

Instead it is about the products the company releases and even more specifically about getting developers excited about playing with them. This only works when you are excited about the product. If the department that builds the product fails to get YOU interested in the product, don't talk about it. Instead, work with the department to find the thing that makes it worth while for you. Anything you evangelize should answer one simple question: Why and how does this product work for me?

The products you evangelize need to get you excited and you need to know where they are going and who to ask for detailed information about them. For example, the product has an internal stakeholder or team that you can communicate with.

Important note: People working on products are in essence not the right people to advocate them: they are far too close to the subject matter to find obvious flaws or are fine with overly complex ways of invoking a certain functionality as they are used to them. Your job is to offer them a way to translate this into easier understandable examples and challenge their current state of affairs.

The real power of removing your brand mindset is that you'll be able to work *with* rather than against the competition.

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Getting speaking opportunities

Like any other skill, becoming a great public speaker means first and foremost doing it a lot. You can read all about the art of public speaking but the main trick is to get in there and overcome your fears and uncertainties.

The crux is that it is not easy becoming a speaker at conferences. This is partly based on the decision of most conference organizers to play it safe and only use well-known speakers. This again is based on the fact that audiences in surveys keep requesting well-known speakers and this is (sadly enough) how we keep the speaker circuit from changing.

There is however a way out, which is to the most part based on getting known and using the great and newer concept of "unconferences".

Going to barcamps

Barcamps are un-conferences, meaning they are gatherings of enthusiasts of networking and presenting. Unlike conferences they are free and one of the really interesting rules of Barcamp is that everybody who attends also has to give a presentation. These presentations are normally 15 minute slots and can be about anything that gets you excited at the moment. Barcamps are a great opportunity to get your first experience in public speaking and allow interested people and conference organizers to spot you as an upcoming talent.

Going to meetups

Meetups are informal gatherings of people in the business to chat about hot topics and generally meet and get to know other local peers. Lately a lot of them have started to have a quick presentation to start the event and this is your chance to get a foot in the door and have an expert audience (or actually an audience of experts that are not the normal conference crowd) to speak to.

Writing articles

A lot of conference organizers also run online magazines. Offer to write articles for magazines and keep a look out for article writing competitions and you'll find yourself invited to speak faster than you think. While in the last years magazines were flush with great content, personal blogging has made it harder for publishers to get good content every few weeks, which means that there is an opportunity for you right now.

Offering brownbags

Brownbag presentations mean that you offer to come to a company during their lunch break and give a presentation. Brownbag presentations are a great opportunity right now:

- They are not that common yet, so offering them to a company makes the person agreeing to it an innovator.
- They target people that normally don't go to conferences but have quite a big word-of-mouth power.
- Your presentation is not disruptive to the daily deliveries of the company but adds extra value to a normal break.
- They don't mean much traveling as you can do them at local companies.
- You get on the company's blog and extranet and via that become known to their clients and partners.

Asking questions at conferences

The final idea to plant in your head is to never be shy to ask questions at any meeting or conference. Success in your job may have its main breakthrough after you asked detailed technical questions to the speakers at a conference. You will most certainly be noticed.

If you don't ask you don't get an answer, it is as simple as that. If your question is technically valuable and interesting you can be quite sure that the rest of the audience had the same problem, but were too afraid to ask.

Media presentation examples

Important note: You must be online to view these examples.

Useful developer evangelist presentation links

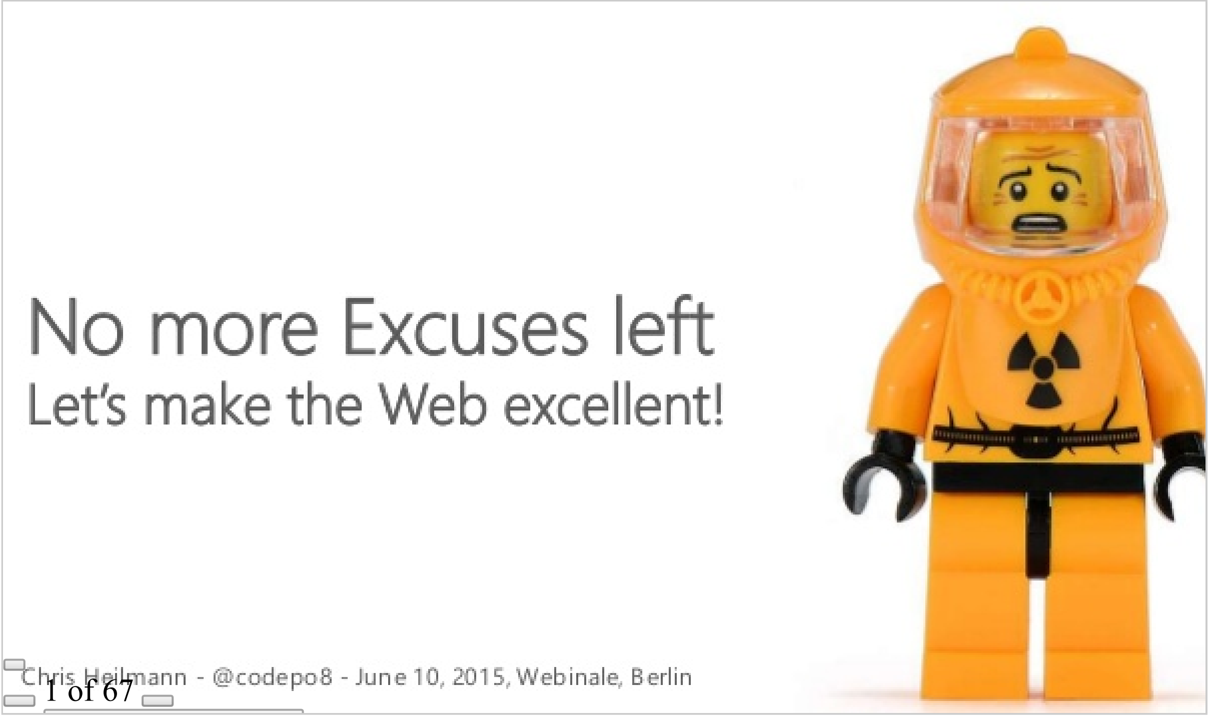
- **Nathan Marz** *(https://www.youtube.com/watch?v=cF8a_FZwULI)* (https://www.youtube.com/watch?v=ucHjyb6jv0)
- **Adrian Cockcroft** *(https://www.youtube.com/watch?v=D3I-GWYB9JY)* (https://www.youtube.com/watch?v=D3I-GWYB9JY)
- **Jay Krepes** *(https://www.youtube.com/watch?v=wKe7w_wOzSI)* (https://www.youtube.com/watch?v=wKe7w_wOzSI)
- **Nick Gorski** *(https://www.youtube.com/watch?v=XwWuUOhL-tA)* (https://www.youtube.com/watch?v=XwWuUOhL-tA)
- **Ben Christensen** *(https://www.youtube.com/watch?v=_t06LRX0DV0)* (https://www.youtube.com/watch?v=_t06LRX0DV0)
- **Lawrence Lessig** *(http://presentationzen.blogs.com/presentationzen/2005/10/the_lessig_meth.html)* (http://presentationzen.blogs.com/presentationzen/2005/10/the_lessig_meth.html)
- **The Lessig Method** *(http://www.ethos3.com/design-tips/the-lessig-method/)* (http://www.ethos3.com/design-tips/the-lessig-method/)

Presentation type examples

The following examples illustrate audio and visual aids you can build for your conference, workshop, and talk presentations.

Slide deck example

A slide is a single page of a presentation. Collectively, a group of slides may be known as a slide deck.



No more Excuses left

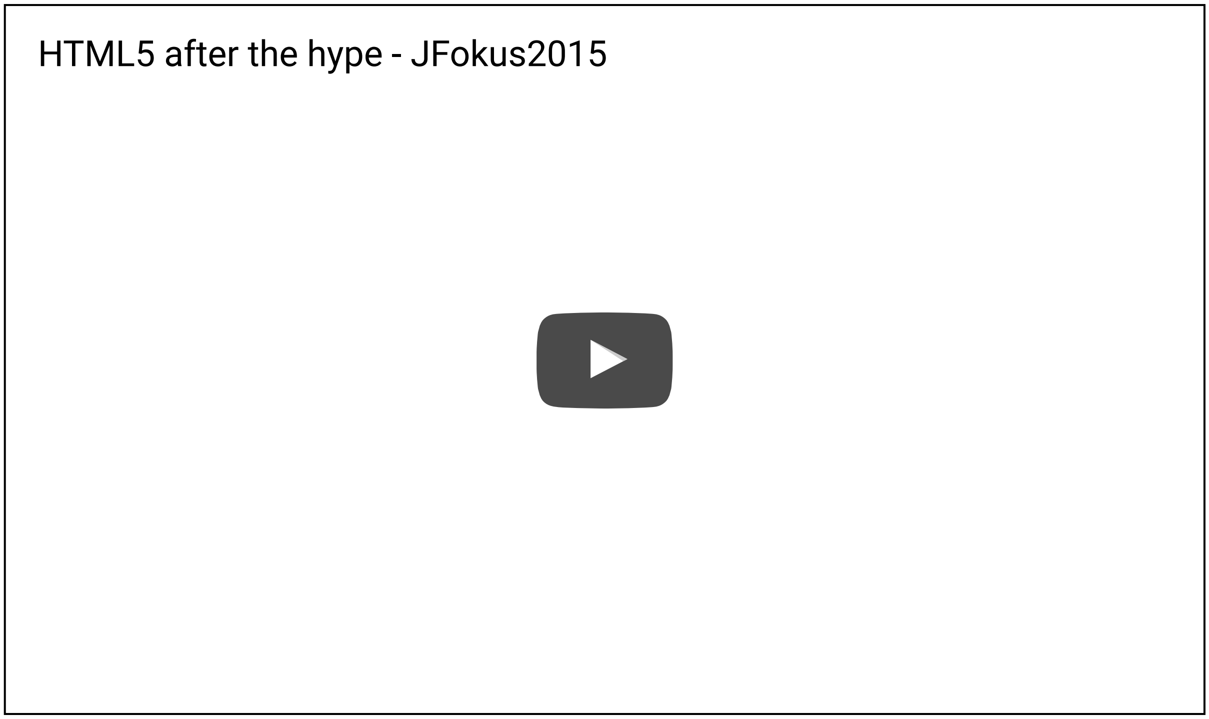
Let's make the Web excellent!

Chris Heilmann - @codepo8 - June 10, 2015, Webinale, Berlin

1 of 67

Screencast example

A screencast is a digital recording of computer screen output, also known as a video screen capture, often containing audio narration.



HTML5 after the hype - JFokus2015

Video example

You can influence wider audiences by pre-recording your presentation and delivering it to many venues.

Why the web is dead | Christian Heilmann | TEDxThessaloniki

Example
Tech Documentation

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Preparing for outreach

As a developer evangelist a lot of your job is going out there and telling the world about the things your company does (or technologies, techniques and methodologies it uses). Your success in this is to a very large part relying on how the world sees you: are you a tech guru or somebody who just tries to sell their company or some product? **Your integrity is your main weapon**. You have to make very sure that it stays intact. This means first and foremost that you need to properly prepare before going out there.

Getting your facts right

You will be asked to talk about a certain new product.

- Make sure that you are up-to-date on the matter before you go and speak about it.
- Do not promise things that are not under your control.
- Talk to the product team and ask them in meticulous detail what the product is about, what works, what does not work.
- Be as skeptical as possible as this is what the people you are about to talk to will be.

Knowing your audience and their needs

Your communication should be targeted to the audience. People came to listen to you or read your article with a personal agenda: if you fulfill that agenda you win. Know what people expect and need and you can deliver. Otherwise you need to hope for the best which is never a good plan.

Going to conferences costs money. Going to free events where you speak costs time. Make it worthwhile for the people who do either and try to get something into your presentation that they can go back to their company with to wow their bosses. That way they will be able to go to more conferences and your other events.

Having expert backup

You cannot be the expert in everything. In the best case, when giving a presentation try to have an expert at hand to answer tricky questions for you. If there is no expert available at the time note down the question and follow it up after consultation. Do not promise to come back to someone and then forget to do so: that will make you look like you needed a fast way out! There are far too many speakers out there who play the "I'll get back to you" game.

Under no circumstances try to wing it and say things you are not sure the product team will be able to deliver. **You are here to promote what can be used**, not to put pressure on your developer colleagues by promising the world the moon on a stick.

Choosing the right medium

Your communication should be in the *right format* for the intended audience. This can range from slides, over videos and audio to live coding exercises or online step-by-step examples.

> A good rule of thumb is: the more technical the audience, the less you should use PowerPoint, etc. Show how you can code with the product, not how shiny it is or what its workflow is.

Planning for failure

Things will go wrong and you need to be prepared. In the case of a presentation do this:

- Have your slides online somewhere in case your local copy dies.
- Have a memory stick with your data on it in case you need to use a computer that is hard-wired into the AV system.
- Prepare to not have your slides available and still be able to do a Q&A session.
- Don't expect any technology to be available. Bring your own connectors, power cables, network cables, etc.
- Don't expect to be able to go online. Bring a stick as backup if you really need to be.
- Aim for a 800 by 600 pixels resolution and expect the worst possible color setting with very low contrast.

Preparing slide decks for presentations

Contents

Overview

Slides are a tricky thing to get right. The main problem with them is that as a developer evangelist you have a technical audience (in most cases) and slide decks are anathema to us. The term "death by PowerPoint" is much more than a Dilbert cartoon. Sadly enough a lot of our day to day life in offices consists of sitting in a room trying to look alert whilst slowly dying inside as some old-school presenter shows us just how many bullet points you can cram into one slide. Again, as with the other chapters, a lot of what you will read here will vary for your experience and environment.

Knowing your stuff

The biggest mistake that presenters make is to rely on their slides as their main source of information. If you don't know the subject matter, or you are not excited by it, or you haven't done much work with it you will give a bad presentation. Nothing makes you a better presenter than confidence in the subject and hands-on knowledge of it.

You will sooner or later be asked to stick to company approved material or "re-use this great deck XYZ has done". Try to avoid this as much as you can. A presentation is **you** telling people about *what you think is important that they hear about*. If you have no clue what the issues with the product are, or if you don't really care about it, you will get into trouble. Technical audiences are amazingly good in spotting what you don't know and will make that the first question in the Q&A session. It is a geek alpha male thing and you will have to be prepared for it.

There is nothing more painful than a presenter turning to his slides and reading out what is standing there. You don't want to become the person that bored people to death. Also, as mentioned in the "Delivering a talk or workshop" section, audiovisual equipment hates presenters and your slides might not be available to you for one reason or another. If you know the subject matter and you are excited to talk about it you will give a memorable talk regardless.

Furthermore, this is you presenting. If the slides are not your style or your language you will appear stilted and you have to remember what the deck says. Public speaking is about giving information in an entertaining fashion: not acting. You should not have to play the role of "corporate speaker" but instead be you. Only then you will be believable and effective. More on this later.

Starting with the content, not the slides!

The first mistake people make is to see the slides as their presentation. The slide deck is an aid to make your presentation easier to take in, and more enjoyable for the audience. For you as a speaker they (the slides) are the narration thread: a reminder of what you want to cover in your talk. A good speaker can keep a room of people interested without any slides whatsoever. A good slide deck, however, gives people memorable moments and information they might miss if they just listened to you.

Starting with a highly portable format: HTML

When writing a new slide deck, start with a text editor. Write the story of your presentation and follow the same rules as for writing online articles. That way you will make sure of a few things:

You know the content and the extent of what you want to cover.
This also allows you to keep to the time limit when presenting.

Have the information in a highly portable format for people to read afterwards.
You can convert it to HTML later on for blogging the notes.

You already know all the links that you want to show and can create easy-to-find versions of them.
For example, bookmarking them in another tool like Delicious *(http://delicious.com/)* .

Don't get carried away with visuals and effects.
This is a big danger and temptation when you play with good presentation software.

Tip: Having these notes makes sure that you will have something for people after the presentation to read.
You can mention this before your presentation and give them the URL.
This immediately relaxes audiences, as the first question at every conference is whether the slides will be available or not.

Choosing a presentation tool that helps you present

Once you know the content, you can start putting together your slide deck.

Choose whatever presentation tool that makes you happy and allows you to simply put your slides together. Keynote and PowerPoint are two good examples.

Also, there are a lot of presentation tools out there that work with HTML and in browsers and that use web standards. Here are some things a presentation tool has to do for you:

Display your slides on the screen regardless of resolution.
Some projectors support 800x600, others go up to 1280x1024.

Easily use, crop, and resize images.
You will use a lot of imagery and it may not be in the right format.

Allow you to freely position elements on the screen.
Sometimes you need things next to each other, and sometimes you need to overlay an URL over an image.

Support remote controls.
You should walk around during your presentation. You should be able to use a remote control instead of hitting the space bar on your laptop.

Have a way to transition smoothly from one slide to another.
This is a subconscious thing but it makes your slide presentation so much more enjoyable.

Be full-screen.
Browser bars or copyright lines and headers distract the viewers.

Have a way to blend things in, one at a time.
This helps your narration and you don't need to repeat content in several slides.

I am sure that given enough time, all of these are possible in browser-based presentation systems too. But why bother wasting time on this when there are perfectly capable systems available i.e., Keynote, PowerPoint, etc.?

Illustrate, don't transcribe

Once you have written your content and picked the right tool, it is time to write your slides. As mentioned before, what you should remember here is that *your slides are not your presentation* but its **outline**. The slides are there to keep your narration flowing and illustrate to the audience what you are on about at this point in your talk.

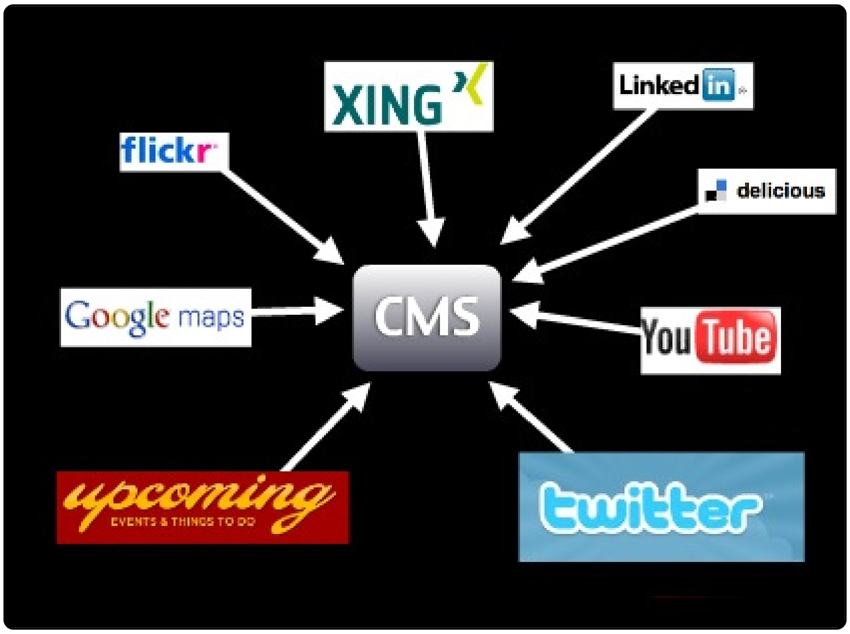
Human communication is to a large part body language, and you standing there and reading from your slide deck, or even worse, turning away from your audience to see what happens on the big screen is communication suicide.

In addition to that it means that you cannot concentrate on the audience. Checking the audience and their body language is a large part of giving a good presentation. It is not about you celebrating yourself and giving a show, but about you bringing information to the audience in an engaging and interesting way. You can only do that when you can see the effect you have: not when you have to read what you want to say.

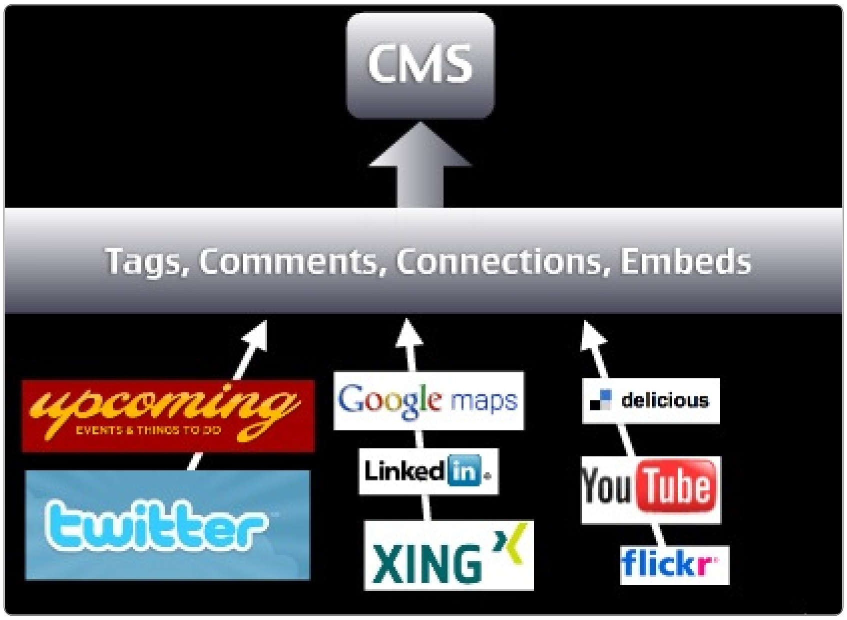
Thus you need to find a single sentence or even a word, a picture, a screenshot, some graph, or some illustration that explains and accentuates what you want to talk about in this segment of your presentation. That way you don't overwhelm the audience with things to read and look at, but you enable them to concentrate on you and you enable yourself to be free in your delivery. If needed, alter your delivery style to stop the audience from nodding off or leaving.

Here is a quick example of information a leading Developer Evangelist included in notes to himself:

> The way to have fun with the web of data is to distribute ourselves around the web and bring the data back to our sites.
>
> The first step is to spread our content on the web:
>
> - Upload photos to Flickr.
> - Bookmark and tag URLs at Delicious.
> - Write short and succinct news updates at Twitter.
> - Upload videos to YouTube.
> - Link addresses and set up driving instructions with Google Maps.
> - Write CVs and bios at Xing or LinkedIn.
>
> The benefits of this approach are the following:
>
> - The data is distributed over multiple servers – even if your own web site is offline (for example for maintenance) the data lives on.
> - You reach users and tap into communities that would never have ended up on your web site.
> - You get tags and comments about your content from these sites. These can become keywords and guidelines for you to write very relevant copy on your main site in the future. You know what people want to hear about rather than guessing it.
> - Comments on these sites also mean you start a channel of communication with users of the web that happens naturally instead of sending them to a complex contact form.
> - You don’t need to worry about converting image or video materials into web formats – the sites that were built exactly for that purpose automatically do that for you.
> - You allow other people to embed your content into their products and can thus piggy-back on their success and integrity.
>
> Here are two slides that were built from data in the preceding notes and then used in his presentation:



Slide 1


CMS
Tags, Comments, Connections, Embeds
upcoming EVENTS & THINGS TO DO
Google maps
delicious
LinkedIn
YouTube
twitter
XING
flickr

Slide 2

Instead of telling all of these things, Chris gave people a visual using the logos of the companies as something they already know, and then showing with a few arrows what he wanted to bring across. He was able to talk through the services one by one and say what people can do with them. The second slide then showed the benefit of piggy-backing on the integrity of these services. Add a practical example of what can be done with this approach and you have yourself a great segment of a talk.

Using and finding images

Images can be a very good way of getting a message across. You've probably seen beautiful presentations with inspiring pictures of swooping eagles and calm waterfalls but this is becoming cliché really fast.

Use images for two reasons:

1. To have something unexpected and fun in your slides.
2. To connect to a real life scenario.

Use imagery to illustrate your point, not to "make it pretty". Pretty imagery might be more distracting than helping and for making it pretty you got colors and typography.

Finding images to use these days is easy. A free and good resource for images is Flickr. Make sure that you use the advanced search and that you tick the boxes for Creative Commons licensed photos that you are allowed to use commercially, manipulate and build upon. The latter is needed if you want to crop the photos. No need for expensive stock photography of multi-ethnic people in suits high-fiving or shaking hands. Tap into and participate in Creative Commons licensing and we all have more interesting slides.

Screenshots are amazingly powerful. Instead of just pointing out a resource on the web your audience can check later, make a screenshot of the web site and overlay the URL on the slide. That way people have a visual idea of what the resource looks like and get a much stronger "Oh I remember this" moment when they visit it. The same applies to interfaces of systems. If you show and explain, you reach more people than when you just explain.

About code examples

Code examples are what a lot of presenters spend far too much time on getting right. You want a good mix of readability and at the same time make it easy for you to change the code. Presentation software by default is not meant for code display. There is no monospace setting, quotes get replaced by "smart" quotes, indentation is all wrong, you have less space than in your code editor and many other problems.

Code examples are however very important as they show people how they can immediately use what you are talking about, and you bring the topic you cover into an area where they feel at home as they use it daily anyway. Showing a few lines of code and what they do in a browser is much more powerful than ranting about the amazing features of the product you talk about. It comes back to the "what is in it for me?" that you should always try to answer with your evangelism.

Tip: Write the code in your normal text editor. Increase the font size a bit and then take screenshots.

This has a few benefits:

- You will have nice color coding which increases readability and helps understanding of the code.
- You will have the right font and code layout, and none of the "magic quotes" annoyances.
- You maintain the code in one spot, and a code change means simply having to create another screen shot.

Provide live demos and source package downloads of your code as explained in the "Writing excellent code examples" section and everybody wins.

Using sound and videos

Sound and video are powerful tools for training and illustrating. For example, a two minute screencast of some system makes it much easier for people to find their way into a system than lots of clever copy (which you'd need anyways cause not everybody can see and hear video).

In presentations, however, it can be more distracting and anything else. Yes, it enriches your slides and teleports you into the post 1990 presentation league (and on Macs it even works), but it is also causes an interruption.

As a speaker you normally want to be the person listened to. You also can use your body language to emphasize the message you want to give. From time to time you point out information on your slides but you bring back the audience to your narration. If you use music or video in a presentation you create a pause as all the senses of the audience are busy following what is happening on the screen. It also creates some time in your presentation where you are part of the audience as you turn and watch the screen (facing the audience while they watch a video is creepy).

Given the disruptive nature of multimedia elements I try to avoid them whenever I can. They look cool but you'll find soon that they are more hassle than they are worth. For example:

- You expect the AV equipment of the location where you give your talk to be able to show video and have audio for you and your computer (good luck with that one).
- You expect the projector to be able to show video. Especially with windows machines and older projectors video is often not visible because of refresh rate issues.
- You make your slide deck impossible to distribute unless you turn it into a video.
- You lose the rhythm of your talk. In essence you create a break that you need to pull the people out of again.

That doesn't mean you cannot talk about videos and screencasts. Upload them to YouTube, take a screenshot of them, and show them as a slide together with the YouTube URL (which you can link and also provide in the HTML notes). That way you can talk about the video and not lose five minutes of your presentation. You can also explain what happens in the video and how it is relevant to what we are talking about. That way people who can't see videos still have an idea what is going on here.

If you really feel that you need to use video or audio in your talk then use it at the beginning or the end of it. That way you either come in as a spectator and become the speaker or get out on the same level as the audience. In most cases video and audio are extra bells and whistles. A good talk doesn't need those. This also applies to transitions and effects.

Don't bling it up

As mentioned before in this guide you can put lipstick and a wig on a pig but it still would make a lousy date. A lot of presenters are happy to use every transition and animation the presentation software comes with, but that doesn't make it better especially when they look bad on a slow computer.

Animation rhymes with moderation and this is what you should always keep in mind. You want to make a point with your presentation and not overload the audience with whooshes and blinking shiny things that distract from your content.

- Use transitions to make your slides smoother.
- Use animation if you want to reveal something bit by bit and avoid having to jump from slide to slide.

Used correctly, animation can be a very powerful tool to make a step by step process more obvious. If you are not skilled in usability and design though, it will most likely appear tacked on and, yes, tacky as animation has been traditionally used to spice up very boring presentations.

The other issue is that animations can actually work against the flow of your presentation. Sometimes you want to speed things up and if you hard-wired long and complex animations you stand there waiting for your slides to catch up with your narration. Sometimes you also have AV equipment that cannot show animations and that makes you wait for something that never happens. One thing you can do is use overly outrageous animation in an ironic manner to show how annoying they can be.

Keeping it brief

Keep your talks brief and if possible cover **one** topic. If you need to cover more than one make sure that you have a good narration flow from one to the other to avoid them appearing stitched together.

Presentations should bring home one message and that one well. This could consist of several sections but the overall story should be obvious. Try to give the whole talk one main theme and return to this in each of the sections.

As mentioned earlier, your slides should contain only what is necessary and not more. There is no point in reading from your slides as that would make you a member of the audience and you have a race who can read it faster.

Try to have either only a theme per slide or one sentence. Try to avoid bullet lists at all costs, especially nested ones. These are old school presentation styles and conjure up unpleasant memories of having to sit through two days of boring training sessions.

An agenda up front is a good idea if you really cover a lot of things, but it also allows the audience to pick their favorites and shut down in between which probably means they miss important parts of your talk. If you do a good job as a presenter the amount of slides is not a problem and neither is at which stage of the overall talk you are. People will be lead through it without realizing it.

The overall amount of slides is only limited by your ability to go through them quickly. A good rough estimate is a slide (which is one topic) a minute, but this depends on how fast you speak.

Considering your audience

One instance where breaking the rules of brevity is when the audience consists of people who do not speak English well and might have a harder time keeping up with your pace and accent. When dealing with an audience like this, having a simple sentence per slide or even some bullet points and repeating them has quite an impact.

- You keep things simple and thus you don't make the audience feel inadequate or that they are missing important things.
- You are forced to pace yourself which is very important with an audience trying to interpret and then understand you.
- You allow for better translation in case you get transcribed afterwards, or have live translation at the conference.

In these scenarios (and especially in the Asian market where asking questions in front of a big group is just not normal) keep your slides more technical. Code is international and people can even repeat it easily and write essays about it in their own languages.

Another thing to remember when giving presentations in different cultures is that pop references and puns do not work. Don't expect the audience to know what you know and to be excited about what you are excited about.

As said at the beginning, a lot of the tips here are for creating presentations for developer crowds. If you speak mostly to designers or management, other tactics have to be applied. All in all it is a good idea to question the classic way of presenting and slide design.

Using corporate and conference templates

During your job as a developer evangelist you will be asked to use conference or corporate slide templates. Try to avoid doing that. The reason is that these templates are almost all the time targeted to the classic presentation style of one heading and 20 nested bullet points followed by a copyright line nobody cares about and other legalese things.

The reason is that using a corporate or conference template is good for the conference and the company but distracts you as a presenter: it is just not you. You and only you should own and run the presentation as it is your integrity on the line. Once a conference or company asked you to be a speaker for them they already trust you to do things right and there is no need to keep the corporate hat on and do a song and dance.

That said, there is a benefit to using these templates. In the corporate case you show a consistent look and feel to the world and align yourself with other publications. The question is if you want that. In my case, not looking like the slide decks of my company battles a lot of prejudices developers have as developers do not trust big brands. This is for you to decide. In the case of a conference looking the same as the others makes your deck more findable later on but at the price of looking the same as everybody else and having a distracting logo on each slide.

The solution is to meet half way. If you make the first page of your deck align with the others and then switch to your own style everybody wins. The cover sheet of your deck is only important in two cases:

1. To fill the screen until your talk starts.
2. As an eye-catcher when you later on send out the deck and show it in a blog post.

Other than that, the slides should take the backseat and only aid your presentation.

Don't reuse without personalizing

Another very common thing that will happen to you once you become a speaker for your organization is that you will be handed presentation decks to present. For example:

- "This has been done by Stephen from the US office and has been signed off by PR."
- "Stephen can't come to the conference, so we want you to fill in for him."
- "Here are his slides, good luck." is a sentence you will hear a lot."

If that happens to you, be firm and say that **this is not how it works**. You are a presenter, not a parrot. If the slides are not in your language, mirror your approach to a certain topic. If you talk about technology or products you are not firm in or have no control over, you are treading on very thin ice. It is **you on the line as the speaker** and the success of your talk stands and falls with how you come across. If you can't be you, then don't do it.

That said, nothing stops you from using the information on the slide deck and translating it to your "language". Instead of flat out refusing to use the deck, say that you are happy to take over and use the information, but that you want to have a chat and information hand-over from the original author. Every speaker has extra information that makes slides make more sense and become more appealing, and you cannot guess these things: you need to hear them "from the horse's mouth".

It boils down to this: giving the talk is one half of the whole show. You will have to answer questions and you will have to be able to explain the practical implementation of a certain technology or product. This you can only do when you played with it yourself and verified your findings with an expert.

Slide decks that get re-used without being challenged and changed become stale. For a company it makes more sense to keep a repository of facts and ways to explain a certain product than full slide decks. That way your information doesn't become stale. Wikis are perfect for that.

Sharing and enjoying

Once you're done with your slides and you are happy with them, don't forget that sharing is caring. Upload the deck to your blog, make it available as a download, or with your company's permission upload it to Slideshare *(http://slideshare.net/)* .

Slideshare is a great tool to get your slides distributed. People can comment on them, share them with friends, embed your slides in conference blog posts, or as a resource for a certain subject and many things more. It is what Flickr is to photos and YouTube is to videos.

Sharing your slides is what a lot of people in the audience will ask you to do and it will get you known as a speaker. People may stumble upon your decks somewhere else and learn about you that way.

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Presenting a talk or workshop

Once you have been invited to speak, got your facts right and prepared for all kind of technical failure you can think about delivering your talk or workshop. This is where you need to make sure you do things really right as data is one thing but data being delivered in an engaging manner is so much more powerful.

Public speaking is an art and there are a lot of tricks you can learn from acting or other forms of performing, but in the end all it boils down to is being prepared and happy to do what needs to be done. Someone has to tell all these people about the cool things you want to talk about, and if you don't do it some sales guy will. There are a few things to think about when delivering your materials. These are described in the following topics.

> There are no bad students or a bad audience: just bad workshops and talks.
> Your mood, dedication and enthusiasm do become those of the audience. If you are not happy, they won't be happy.

Being yourself

You will find dozens of books and videos on how to be a great presenter. However nothing makes you a better presenter than being who you are. You should not have to play a role or dress up. If you believe in what you do, you will be great. Your best asset is your confidence in your product and in yourself.

Confidence does not come naturally, but it will get easier the more you present. Prepare your materials and expect everything to go wrong and there won't be any bad surprises.

If things go wrong, and they will, take them in stride. Say flat out when you made a mistake and get on with it. One of the main things to do is to give the audience the impression that while you are an expert you still are a human being, prone to error just like everybody else in the audience.

Invite communication

All of what you do as an evangelist is about **communication**. You are a sender that brings a message to the audience but you are also a receiver that brings the issues, concerns, and ideas from the outside world back to the company.

If you give a talk tell people that it is OK to ask questions. Make space in your presentations for that. Stop after a complex part of the talk and ask the audience if all of that was understandable or if you should repeat some detail. Ask them questions and have small presents for people who answer.

It is generally a good idea to ask the audience questions from time to time. Ask them what they do (show of hands) and if they had experience with the product already and so on. This will give you an idea how to pace the rest of the talk but also makes people feel that they participate and keeps them from nodding off.

Prepare takeaways

People should have the chance to concentrate on what you are saying and shouldn't feel that they have to jot things down to keep up.

- Have a URL where they can download your information afterwards and show this as one of the first slides.
- Have all the links in a presentation as a tag on delicious (or any other social bookmarking site).
- Say upfront what you will cover and what they will get out of it.

Making people guess makes them feel uneasy and that is not what people should feel like when they listen to you!

Plan time for and own the questions and answers

Plan time for a "Questions and Answers" session after your talk. Most conference organizers will do that anyway, but be generous. These sessions allow people to ask exactly what they need answered and go back and have a go at immediately solving their problems.

One thing that is very important is that you need to be in control of the Q&A. A lot of times you will have people who don't ask questions but profile themselves instead. Deal with that accordingly and swiftly. People will have real questions that need answering.

If you find a person in the audience that talks for a minute about their skills before asking a question cut in. Ask the person for their name and single them out as an expert. "Great, Steve here knows a lot about issue XYZ, so during the break you can ask him about issues dealing with this. Steve, let's collect some of the questions around this and work through them later together." That way you gave Steve a chance to deliver his knowledge, made him feel like a million dollars, and another person has a chance to ask a real, burning question.

Be honest and real

If you don't know an answer, do not speculate.

Instead a great message to give is to ask the audience if someone knows. Normally there are other speakers or "silent experts" in the audience that can help you out. That way you show that you are also open to learn, and the stigma of the "arrogant speaker" is broken.

If there is no answer, offer to investigate further and swap contact details with the person who asked the question. There is no harm in not knowing something. *Though there is harm in lying.*

Using follow-up communication

Whatever you do – it is important to cuddle afterwards. In the case of a presentation this means that you should make sure you email everyone who gave you a business card (which can become time consuming but is something you can do on a train). Make sure to blog, upload recordings, photos, and slides as soon as possible. This shows respect to those who came to see you talk, and invites those who missed it.

Have contact options available after your talk (normally on the last slide): email, twitter name, and so on. The best plan is to have communication channels that are not your company mail or IM name you use at work. For starters this allows you to be more selective in answering but more importantly giving out company communication channels to anyone can be a security risk.

Additional presentation tips

Here are some tips to use when giving presentations.

Introduce yourself

Introducing yourself, however briefly, breaks down an initial barrier. You are not any longer this unreachable person on stage or at the head of the table: you are a normal person. Explain why you are competent to talk about the matter at hand. Then put the ego away: people came for information, not to see you sing and dance.

Use humor

Humor is important to keep a long presentation interesting. Put in things that people just don't expect to keep both you and them on the ball. Humor also makes things more approachable. We tend to use humor to deal with things that scare us. Furthermore humor allows for a memorable moment: it is a way of structuring and providing *landmarks in your presentation*.

Build bridges to the real world

Bring up real world examples and comparisons. The rationale is that they make very theoretical and hard to grasp data more easy to consume for humans. Real world comparisons also allow for emotion: emotional responses are very powerful and make us remember.

Example: If you for example talk about code standards and re-use of code *without a proper review*, a good case to mention is the Ariane 5 disaster. This rocket self-destructed because it veered off its intended path 37 seconds after lift-off. The reason was re-use of the code used to launch Ariane 4, which had different flight specs. 370 million dollars were lost due to this error.

Pace yourself

Speaking at the right pace makes you easy to understand. If you appear rushed, listeners will feel uneasy. Trying to keep up is a terrible feeling and makes us feel inadequate. So speak slowly with meaning and concentrate on thoroughly pronouncing things. Pauses are good. They allow listeners to take information in and digest it in the way they know best.

Avoid "Hello World"

"Hello World" examples are easy to show. They are also useless, as they teach a syntax, but not the concept of a language or solution. There is no personal value in "Hello World". We should teach how to solve issues and fulfill tasks. It is much better to have a real production example to build upon:

- "This is what we had to create: here are the specs."
- "This is the final outcome."
- "Here's what we used to deliver this job."
- "... and here is how you can do it yourself!"

Build on top of what people are asked to do, not what you expect them to do for you.

Be fresh

Always try to deliver fresh material. The least you can do is to bring some new, fresh angle. Check what is hot at the moment, research it and add it to the talk. That way you show that your content is not only good but also very relevant at this moment in time.

It also means that you don't bore seasoned conference attendees with something you told them before. Seasoned conference attendees are also adamant bloggers and Twitter users so this can only be a good thing.

Recording your output

Making recordings of your talks and in general your work is a good idea because of several reasons:

People who couldn't attend your session get the same information.

A presentation is much more than the slide deck. The deck is actually just the table of contents of the talk.

You have the chance to check how you come across to the audience.

You can check your talks on your iPod while cycling in the gym. It's a good use of time and you can see where you need to improve.

People who prefer audio or video get interested in what you do.

This also includes people who need audio and video because of their condition. For example people suffering from dyslexia, low vision, blindness, deafness, etc.

It allows you to publish in other channels.

This includes your blog, site, or conference archive.

Recording things is dead easy nowadays. You just need to know what to use and where to put it.

Recording the audio of your talks

Most conferences will give you a lapel mike and do some proper audio recording, but if that isn't the case or you simply want to have your own copy, the built-in microphone of a MacBook Pro is perfectly capable of making a good recording of your talk if you don't walk around too much. The other option of course is to get a small external microphone.

For recording, try Audacity *(http://audacity.sourceforge.net/)* which is a free, open source sound editing tool that has all the capabilities you need (record, cut, convert). Once edited, put the file into iTunes to tag it and add the images for the coverflow for iPods.

For storage of audio files, try archive.org *(http://www.archive.org/index.php)* which has a pretty nice uploading tool and comes with a good search functionality and in-built player.

Audio recordings are great as they are comparatively small and people can put them on their portable music players and listen to them on the train to work. In addition to that, SlideShare *(http://slideshare.net/)* also allows you to add sound to your presentations to make them slidecasts. The editor is easy to use and it gives presentations an extra zing.

Shooting video

Having a video of your presentation or interviews is very nice, much for the same reason of having an audio recording. As there is audio and video it does give people the full experience of seeing you talk. People love to get videos and use them as training materials.

You can get small cameras with amazing recording quality these days and even smartphones allow you to record. Editing is a bit more complex than editing audio, but I found that using mpegstreamclip *(http://www.squared5.com/)* makes it easy to do simple cutting and conversion into almost any format.

Hosting is a bit harder. archive.org is again an option but the real power of online video comes from hosting it where people are used to looking for videos and can embed them in their own blogs. YouTube *(http://youtube.com/)* is of course a main candidate (also because of their great annotation tool), but Vimeo *(http://vimeo.com/)* is also good. Uploading takes a lot longer than the smaller audio files so it is pretty tough to have a really quick turnaround.

Making screencasts and screenshots

Another really powerful tool to show people what you are doing are screencasts and screenshots. Sometimes a picture explains what you want to achieve much easier than a bunch of instructions. Step-by-step instructions on how to use a certain interface (for example how to sign up for a developer key) are very easy to show as a screencast. Describing the interface with words is much harder: just try to explain to people on the phone how to install Windows for example.

Try keeping screencasts very small by just filming yourself going through some interface. You can also jazz them up with voice-over or embedding your webcam.

By far the easiest screencast software for Mac is ishowu *(http://www.shinywhitebox.com/home/home.html)* which is not free but dirt cheap. If you're on PC and have some more money Camtasia *(http://www.techsmith.com/camtasia.asp)* is another tool to use.

For screenshots Skitch *(http://www.skitch.com/)* is very powerful, especially as it allows you to annotate, highlight, add arrows, and immediately upload it to the web.

Building link collections

Another great way to record what you have done is using social bookmarking (like Delicious *(http://delicious.com/)*) to collect links for a certain event or talk that you've given. Instead of people having to remember all the links you've used in a certain presentation all they have to memorize is a single URL with a tag. That way you can easily find interesting URLs you talked about later on: simply get the link collection with the presentation tag.

The other benefit of this is that people can tag and add notes to your links in the social bookmarking system, thus making them even more findable. This brings us to a very important part of your job as a developer evangelist: using and knowing the web and the social web.

Example
Tech Documentation

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Travel and conference attendance

Contents

Overview

Conferences and travels are stressful, no doubt about that. They can also be frustrating. For you as an evangelist they are work, and as a great evangelist you should take them serious. Every one of them. If you can't take an event serious, don't speak there.

As with everything about evangelism, your conference success stands and falls with your excitement. You will have prepared your talk and researched the audience. You should have looked at what other speakers are doing and talking about to avoid overlap. That's why it can be annoying when others use the conference as "time out of the office". And it gets worse when organizational problems mean you ended up in an almost empty room. Try to avoid this. Be as excited about the event as the organizers and attendees are about you being there.

The fact that every great evangelist should always keep in mind is the following:

> If you are invited to speak at a conference, you should be present at that conference.

People paid money to see you and the organizers rely on you to entertain, inform, and educate. If you are not present at the event, you are cheating both the organizers and the attendees. You are not a rock star who can sing one song and then leave covered by body guards. You need organizers and attendees to do your job. This starts by planning your trip to the event the right way.

Getting your travel and accommodation sorted

A whole book could be written about this. Of course, things are different from conference to conference. But here are some points that are important that will help you keep your sanity during all the time on the road.

Book your travels with at least one buffer day before and after the event.
: There is no way you can be a great presence when you parachute in. Have a good night's sleep beforehand. Conferences tend to start early, and you should be awake, bright and bushy-tailed. There is a whole day ahead of you, if not two.

Stay close to the venue.
: While you can save money by being in places somewhere else it means extra commuting time. Also, as you are not likely to know the city, commutes can become confusing. Conferences in hotels are the ones most travel-friendly and you can book a room there. This means you can travel light to and at the event with just your laptop bag and some cables. There is nothing more annoying for you as a presenter to have to carry lots of layers of clothing and luggage around with you. You are there to represent, not to move in.

Travel in affordable style.
: Flying is stressful, recycled air is bad for you. Every other person at the airport is a potential obstacle which might make it impossible for you to arrive on time. Ask for a good mid-price ticket and know your airlines and airports. Make sure you sign up

for a frequent flier program. The more your advance in these, the cheaper it gets to get tickets as you can upgrade yourself with air miles.

Conference organizers are the best people to ask about their event regarding travel and accomodation.

Who pays what?

This can be a controversial topic. It is also an important topic. Conferences are like concerts: without a band you have no concert. This means organizers should treat you well and you should treat them right. Nobody likes a diva, no matter how good you are. Nobody likes a cheapskate either. If the conference budget has no money for at least accommodation and travel of the speakers, it is a terrible budget. You being at an event means you give your time, dedication and effort. That should be recognized and paid for. If you travel on your own expenses, you effectively pay for the event you speak at. This might be necessary at the beginning and if the conference is a massive opportunity for you. In the long run it is neither professional nor maintainable.

You are likely to work for a company if you are an evangelist, and the company can cover some of the cost of your event participation. This can also be part of a sponsorship package: we pay our speaker's expenses and get coverage. But for you as the evangelist there are a few dangers in that.

In essence, you need to be you on stage. You don't want to be a shill that represents a company because he or she has to. This means you should be somewhat independent of both your company and the conference organizers. Neither should be able to tell you what to cover but instead trust your professionalism to do the right thing. If that is not possible, you need to work on that first. It is trust going both ways. Many companies will want to pay for your presence so that the organizers can't tell you what to do. This is nonsense, as it just means your company wants to be in that position.

There are many facets to this, but one thing should be high on your list:

Avoid any sponsored speaking slots! These are not for evangelists.

These are for sales people, and nobody wants them. Audiences are annoyed when something is obviously a veiled advertisement. Presenters know people expect them to give them hot air with glitter in these talks. Sponsored speaking slots need to die in a fire. They are a product of a bygone time and are only around because of convenience.

In the best of scenarios try to aim for the following:

- Have the conference organizers cover your travel and accommodation. After all, they know best.
- Repay your company by representing them and by delivering a report after the event. How was your talk received? What did the competition do? What contacts do you have to follow up on? Did you meet a prospective hire?
- Split the difference. Your company can cover the extra days in the hotel and expenses on your travel. That makes it cheaper for the conference organizers.
- Have a clear separation between your presence at the event and any form of sponsorship. You should not become something to barter with. You should pick the events you go to and not have to go as a speaking slot was part of the package. This is insulting to anyone involved and will neither help you nor your company.

This will be something you have to define rigidly and fight for. Being a speaker that paid for getting into the conference is reputation suicide. Do not try to get into that position. Your reputation is that of being an independent technical person, not a marketeer.

One other thing to keep in mind is that a lot of people present at conferences for a living. For them not getting paid to be there means they lose money. Don't be the person that under-bids them and make enemies that way.

Be at the event

It is very tempting to go, deliver your talk and meet friends or go and hang out with other speakers doing some sight-seeing or shopping. It is also not respectful to the attendees and organizers. Plan these activities around the conference, not on the day of it. Many great conference organizers do that for you. There is normally a speaker dinner before the event and some organized city trip afterwards. If you come across conference organizers like that, thank them. These are the good guys.

Try to be at the event when it starts and take part. Look at other presenters and see what they are doing. Talk to the people at the booths there, you know how boring that job can be. The more you soak up before your talk the more you can bring in later. How cool is it to remind the audience of something you heard in another presentation at the same event? How happy are sponsors when you mention what they show on their booth? Remember, the money the organizers got to get you here came from the sponsors. Pay some of it back in kind.

Give the event some social media love

You are most likely a voice on the social web already, or at least try to become one. Conferences are a great way to get a boost and get more people to follow you. This is pretty simple:

Have your social media contact info on your slides.
Introduce yourself with them. It is pretty amazing how many more followers you get by this, both at the conference and by people who read the slides online.

Cover your presence at the conference.
Tell people when and where your talk is using the conference hashtag. Tell people where you are during the breaks in case they want to talk to you.

Point out fun things and good talks by others.
This is investing in the future. If you share your excitement about other people's work, they just might share theirs.

Retweet conference organizer updates.
A bit of advertising doesn't cost you anything and they'll be grateful.

Have your materials online and tell people about them.
Your slide deck/demo materials/videos/new product release with the conference hashtag attached to it will get much more people to find it and use it. For people at the conference this can also be a "try before you buy" opportunity.

Use the event to build a network

It can be daunting to come back from an event with a whole stack of business cards. Especially in a day and age where we all have smartphones that would allow us to immediately contact one another. Don't be a "grab and forget" person though. If you got a contact, follow it up with an email. A lot of time nothing happens afterwards, but at least you have the details in your email client already in case you need it later. I started taking photos of business cards and stash those instead of the cards themselves. I shred those - it is not cool to throw away people's personal info like email and phone numbers.

Networking can be fun at events. It also can be very beneficial at times not to hang with the loud and rowdy crowd at after-parties and networking events but keep a bit to the outside. The people who are scared of the loud crowd will find you that way. You can build more interesting contacts and conversations that way.

Keep track of your conference participation

This can become a full-time job once you get known and there is a lot of demand for you. Make it easy for yourself by having a good calendar where you track your events. Follow up each of your conference participations with a debrief for your colleagues and for the conference organizers. This makes it easy to decide next time if the same event is worth you going or someone else. It also gives organizers something to think about and you a chance to say "thanks", and that goes a long way.

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Using the conference buzz

Speaking at conferences is a great thing to do, especially as there is a massive opportunity to network with other speakers and get to know what they are up to. A lot of misunderstandings with different technologies became a lot clearer after a few beers with people from the respective teams.

Conferences also create a lot of online buzz and are a great channel to get your information out to lots and lots of people. This is to a degree dependent on the size of the conference. Smaller ones have less buzz but something like SXSW might have too much and your contribution will get lost in an avalanche of tweets of people trying to find and meet at a certain bar in Austin.

The trick is once again to come from a different point of view. In addition to following the normal marketing procedure of conferences, try to find the extra "what is in it for me?".

Being a part of the conference

Organizing conferences is quite a tough job so a nice thing to do is support the conference you speak at. Twitter about it, tell people you'll be there, maybe organize a small informal breakfast or dinner meetup in the days and hours around the conference.

The main benefit of going to conferences, regardless of being a speaker or attendee, is to mingle with others and exchange thoughts, ideas and information during the breaks and before and after the event. Don't just show up for your talk and leave: you'd miss out on most of the fun.

Release immediately

During a conference and in the days to follow the web is a-buzz with tweets, blog posts, photos, links and all other kind of goodies with the tag of the conference. Conference organizers also start to show twitter updates live on the big screen and collect web content tagged appropriately to list on the main conference web site.

This is a great opportunity for you to get your stuff out as far and fast as possible. Have your slide deck ready on SlideShare with the right tag and put it live immediately after your talk. Twitter about it using the hashtag and add the conference tag to the links you put on social bookmarking sites and you'll be part of the first wave of information.

The same goes for your photos. Upload them to Flickr or Facebook, tag them appropriately and people will find them as everybody checks the conference photos. Make sure to tag photos with the name of the people in them to make searching even easier.

Writing about the conferences

Another good way to give back to the conference is to cover it in your communication channels. Write a small blog post about your session at the conference (of course), but also a general post about the conference and what you liked about it. Also give personal feedback via email to the organizers after each conference and you may get a lot of thanks for that.

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Using the social web

Contents

Overview

The web is your biggest weapon as a developer evangelist. It is a world-wide, 24-7 information and communication channel and allows you to get your message out.

By understanding and using the social web you will also find that other people become relays for your great stories and that people will even translate and publish in their own markets for you.

Social media is an amazingly strong buzz word these days and a lot of money is being made by giving "social media consulting" or even "Twitter workshops". While the tricks told in most of these will give you instant success, this success will also fizz out faster than you can say "hype".

A lot of the "social media experts" sound a lot like the "search engine optimizers" of the recent past, and use the same dirty tricks.

Be aware of that. Your job is to use the web as a communication and distribution channel: not to make a mint in a week selling snake oil. This would kill your reputation and as stated before your reputation, integrity, and honesty is what makes you a developer evangelist. Losing these will make it almost impossible for you to get listened to in the future which renders you useless for your company and gives the general idea of developer evangelism a bad name.

Finding great web content

As the web is where you want to publish, you also need to be interested in it and find great things to tell people about. Also, you cannot exclusively talk about your products.

The most time-efficient way to find content is to collect yourself some good RSS feeds in an RSS reader. Google Reader is a good example.

Other great resources are Del.icio.us *(http://delicious.com/)* (as people tag and describe technical content really well), topical mailing lists and forums. Don't just find things: make sure that what you find was ratified by a human or humans you trust. Then you can safely re-distribute it.

Redistributing web content

Once you found great content, re-distribute it. The reason is that you have a different network than other people have and you should never assume that people already know about the things you find.

You can re-distribute web content in several ways:

- You can blog about it.
- You can add it to social bookmarking sites and add a good description and tags.
- You can use it in a presentation.
- You can quote it in a mailing list or forum to add more relevance to your post or mail.
- You can Twitter about it.

In any case, the most important thing is that **you attribute the content to the originator** by name and resource. The reason is simple: clever web users track what people do with their content, so if you blog about a subject, your blog will show up on their radar. The same works for re-tweeting. In other words, you get known to them which could be the start of an interesting two-way communication.

Being known on the web

If your job is to bring interesting news and explanations about web products to the web then it should be pretty obvious that you should not be a stranger on the web.

Sign up to mailing lists, post on forums, use Twitter, poke around IRC channels, leave comments on interesting news articles of tech magazines and online magazines. Simply don't be shy to give your point of view or real advice whenever you can.

Be aware of new and upcoming networks and social apps and sign up for them as soon as you get the chance.

Being visible is especially important when you work for a large company. Tech news portals love to bring news about big companies, especially bad news. It is also stunning how many times these news items contain distorted if not outright wrong information.

As the marketing and PR departments of your company are most likely not aware of these tech publications and hardly ever would step in to rectify the mistakes, in their reasoning this is a good chance for you to act. Stick to technical, real information and show proof of your points. Of course some people commenting will side with the misinformation as it is just cool to stick it to the man and fight large organizations, but the lesser vocal majority will at least get the real story straight from the horses' mouth. Be very sure about your facts if you do that though!

Social media experts and entrepreneurs will tell you that it is terribly important to use your real name and have a domain with your real name and make it all very personal indeed. Sometimes a handle that is more technical or geeky actually gives you more credibility and makes people listen more, especially when you used it for years in other circles like old mailing lists, usenet or IRC.

Using powerful social web sites and products

The social web is evolving almost monthly and especially right now there are a lot of companies being shut down or others emerging in weekly cycles. This is OK: it is an evolution after all. For you as an evangelist this can be very interesting:

Write about new social media products.
Have a play with them, note your first impression.

Be on the lookout of new products.
If you sign up quickly you normally get invite codes (a very common practice in launches), and having those invites as one of the first and saying so on Twitter is way cool.

In addition to the new kids on the block of the media space, it is very important for you to be aware of great resources that are already established, have a network of specialists, and allow you to easily store and access content:

- Flickr *(http://flickr.com/)* is a photo sharing web site that allows you to store photos and screenshots. A lot of tools can talk to it via its API and it comes with a bulk uploader. Flickr is also a great resource for Creative Commons licensed photos for your presentation. Users can leave comments, tag your photos (add keywords to ease finding) and leave notes on the photo itself. Flickr also stores short videos (long photos as they call it). It is a massive community and a great way to interact with people.
- YouTube *(http://youtube.com/)* is the #1 video sharing web site in the world, run by Google. It is very easy to store video on YouTube and embed the videos back into your blog or web sites. Users can comment and tag. YouTube also has a great API to use the video and create your own custom players if you want to. The support for annotations and captions makes it very accessible and allows you to easily add extra information to video content.

- Vimeo *(http://vimeo.com/)* is another video sharing site, which does the same as YouTube but has higher quality content and much better usability. It does take a while to convert video though.
- Archive.org *(http://archive.org/)* is the Internet Archive which allows you to easily store videos, audio and pictures that you want to release to the public.
- Dopplr *(http://dopplr.com/)* is a social travel site. You can store there which trips you make and find and notify people thus when you are in their area. You can also find out when people are in your city and for example invite them to the office for a talk or interview.
- Del.icio.us *(http://delicious.com/)* is a social bookmarking site - users can add notes and tags to your bookmarks and share them across networks.
- GitHub *(http://github.com/)* is a social code sharing network. Git is a version control system and GitHub allows you to store code there, have a Wiki to explain the information, and make it easy for other developers to fork and watch your code. GitHub also comes with a nice code displayer and automatically creates archives of your code for people to download so you don't need to zip it up after each change.
- Google Code *(http://code.google.com/)* is another way to store your code for people to download.
- Google App Engine *(http://code.google.com/appengine/)* even allows you to store and execute your apps on Google's servers.
- Google Reader *(http://reader.google.com/)* is not only a great RSS reader but it also allows for sharing with networks, adding notes to each other's subscriptions and tag content.
- SlideShare *(http://slideshare.net/)* allows you to store presentations and offer them as Flash embeds to the world. It allows for tagging and commenting and SlideShare even allows to add a sound file to the slides to turn them into a slidecast.
- LinkedIn *(http://linkedin.com/)* is a professional network where you can find other evangelists and key people in companies you want to reach.
- Facebook *(http://facebook.com/)* I am sure I don't have to explain. Good for event organization and contacting people quickly, less useful for photo storage because of their terms and conditions.
- Upcoming *(https://www.kickstarter.com/projects/waxpancake/the-return-of-upcomingorg)* and Meetup *(http://meetup.com/)* both are social networks revolving around real-life meetings and events.
- Twitter *(http://twitter.com/)* has taken over the online world by storm. It is still re-inventing itself daily trying to define what it is, but one thing is for sure: it is an amazingly easy way to spread short information very far and very fast and a good way to stay in contact with people when you are on the go.

These are only a few of the sites that make a lot of sense to use as a developer evangelist.

> **Tip:** I found that every successful social network does not have the network as its main core idea but revolves around a thing people get emotionally attached to: pictures, videos, travel, music, and so on. Pure networks hardly ever keep one's attention for long.

As with anything, these resources are as useful as you make them. Good title writing, describing, annotating, and tagging will make them more useful to the world and easier for you to find what you've put up there weeks ago.

Using the web for storage, distribution and cross-promotion

The web is a network of products, documents, data, and sites and to use it to its full advantage make sure to spread your content around it. Having a product in one place is great, but people have to find it. If you put parts of the product in different places which were designed to host a certain content type, you give web surfers a lot more opportunities to find your content.

> **Tip:** Don't forget that social networks can be very insular: not everybody has the time and dedication to be active on lots of them. Therefore having information useful to the nature of a certain network pointing to a full product elsewhere makes you a part of the network but also gets people interested in following your link and checking out what you're talking about.

Spreading your content across different platforms has a lot of benefits:

Multiple points of storage

Even if a very successful blog post would kick your server off the grid (slashdot or digg effect), the information on the other platforms is still available.

Multiple feedback channels

People can comment or inquire where they are happy to hang out, may it be YouTube, Flickr, or Slideshare. All of these web sites spend a lot of time building close-knit communities which means more expert feedback for you.

Automatic conversion and hosting

Flickr specializes in hosting and converting photos and short videos, YouTube does the same for longer videos, and other specialized sites know how to tweak their servers to convert and send their specialist data over the wire the fastest which is something you don't need to think about.

> **Example:** Say you have a new code solution. You can write a blog post explaining it, put screenshots on Flickr, a screencast of how to use the interface (or install the solution) on YouTube, have a presentation about the solution on SlideShare and host the code on GitHub. This quintuples the potential audience just by using all these systems in the way they were meant to be used.

Spreading the content is one part of the solution that will make this a success for you. What you need to make sure of is to link all of these bits of information back to the main product. Write great descriptions for videos and screenshots, use informative tags (one of which can be the product name) and track the feedback on all the channels to be able to answer people's questions immediately. There is not much point in a multimedia resource on the web when nobody knows what it belongs to.

Hint, tease and preview

Social media is a lot about exploration, and showing off just how much better you are in finding new information and learning about new products than anybody else.

> **Fact:** This is a general thing about the web: people have gigabytes of information already downloaded, but rather than consuming this information we spend most of our nights hunting for more and adding to the big pile of what we already have. It is a human thing and deep in our psyche: ever since we started hogging food for the long, cold winter months that we had to stay in the cave.

You can use this to your advantage by previewing, hinting, and teasing about upcoming releases. You can do this a lot: not consciously to manipulate, but because you are too excited about things and prone to release information before it can be publicly available. Instead of releasing too early, which is disastrous, you can doing some of the following:

Upload screenshots of upcoming products to Flickr.

This results in pretty cool comments and people tagging the photo with keywords you may not have thought of adding as content to the product docs.

Upload screencasts to video sharing sites.

The same reason, but bigger impact. However, remember that watching a screencast expects more buy-in from the consumer than just looking at a screenshot.

Hint about cool stuff coming up on Twitter.

This will result in a lot of your followers directly messaging you asking for more details, and you can give out previews and insights for them to be the first to talk about it.

Flat out ask for beta testers.

People love poking at things before they are public. Feedback achieved this way can help you spot sources of confusion you weren't aware of, and allows you to fix them before release.

Once the product is out, don't forget to add the real URL to the previews you already spread on the web. That way all the late-comers will know where to go.

Tracking your impact

There is not much point in putting your content out on the web without knowing what your impact is. You can track the effects of your publications in several ways:

Add a page counter.

The thing to read there is not really the amount of hits and get high on that, but to **look through the referrers**. I found a lot of cool blogs in my stats. I am using StatCounter *(http://statcounter.com/)* and Urchin which comes with my MediaTemple *(http://mediatemple.net/)* server.

Search the web and social media for keywords.

Twitter search *(http://search.twitter.com/)* and Google Blog search *(http://blogsearch.google.com/)* are great for that.

Subscribe to comment feeds

These are of posts you've written or commented on.

None of this is rocket science, but it can be very powerful and teach you how successful different styles of publication are.

Building a network

As the web is much more social these days than it used to be, it is easy for you to connect to other people. It is very easy to get up-to-date information from people on Twitter, Facebook, and LinkedIn and you can give information to the right people with the right networks almost instantly.

You achieve this by contacting people, re-tweeting cool things they talk about, report problems you found with systems on Twitter and other similar tricks that really are not rocket science.

People who really grasp the social web are happy to answer to you once you've proven that you have interesting things to say, and that you find and re-tweet, bookmark or blog about great content. All of this is to make our research time shorter and our lives easier. Other evangelists are by definition social people, so don't be afraid of talking to people and pointing them to things that get you excited.

Once you've successfully done that you'll find that you get good information earlier and earlier, and before you know it you are the one with the invite codes to private betas and know and talk about cool new stuff that is in the making.

Example
Tech Documentation

Developer Relations Evangelist Guide

Created: **2011-01-10**
Security classification: **PUBLIC**

Working with your company

Contents

Inside your company

As a developer evangelist you will find that a large part of your work is dealing with internal developers and unpredictable changes to your company.

This can actually be a much harder job than dealing with outside developers. The reason is that people on the inside do not only see the outcome of the products and may disagree with these outcomes, but they are also part of the journey towards this result. That journey can be painful, frustrating and sometimes just plain confusing.

Your job as a developer evangelist is to:

- Listen to your developers.
- Understand their problems.
- Communicate with management to try to sort the issues out.

> You also need to ensure that you remind people of facts instead of chiming in with bad rumours.
> What the outside world says about your company and its products may not necessarily be the real picture.

There are a lot of things you have to be aware of when you are a developer evangelist. One of them is the enemy within the company.

Preparing for prejudice

You have a very unique and new role which spans various traditional roles. This is especially true for developers moving away from the day-to-day grind that can easily be seen as betraying the cause. Prejudiced developers are amazingly proud of being the delivering part of the company and consider anyone who does not code superfluous. I am sure you met these people and heard statements like “I don't know why we need designers, I know how to use Photoshop, too!”. It is somewhat ironic that the people whose standing in the company you try to improve are the ones that are likely to be opposed to your role.

At the end of the day, this is something you will have to live with. You may burn a few bridges and you will have to suffer a lot of internal and external abuse and nagging, *but it is important to keep your eyes on the goal*. If you have been working for 12 years as a developer then one thing is true: delivering awesome work pleases first and foremost yourself. In order to have an impact in the company and to get things changed you have to take your hands off the keyboard and start talking and persuading.

Don't get discouraged by people seemingly stabbing you in the back. In reality this is exactly the miscommunication and lack of people skills that you try to help out with. See it as a challenge and not as a show-stopper. Once you can show successes in improving the standing of developers in the company you can go back and call a comparison match. Most of the time this is not necessary. The terribly grumpy people either left the company or others start telling them to stop.

Dealing with company changes

As the software market is constantly changing so do the companies in it. This can be a great thing but in most of the cases it will mean that you have to deal with happenings that are beyond of your control. Mergers, acquisitions, new products, products being discontinued, rounds of layoffs, redundancies, etc. All of these things happen, so be prepared.

The thing about you being a developer evangelist is that you are in the spotlight of the world and the company so you'll be the first to be asked about an opinion when there is a big change in your company. This can have disastrous effects: either you violate a company policy and tread very hard indeed on the toes of your Legal, HR, Marketing and PR teams or you sound like a company drone to the geeks in the company. Here are a few things to be aware of:

Every change in the company has a legal process.

While you are not an official press channel to the outside world, your statement can be misquoted (oh and it will) and get you into legal trouble with your own company. Liaise with PR and Legal as soon as possible when there is a change. Then tell the developers in your company the legal implications.

There is no "off the record".

Neither internally and especially not when you are talking to the press/bloggers/outbound channels.

Switch to listening mode.

During the first few hours after a dramatic change, listen to everybody and keep your eyes and ears open to what people say. This helps you stop people from completely destroying their professional standing in the company and the market and learning about what is really happening. Don't be part of the noise that drowns the facts.

Don't act emotional and make assumptions.

Almost any change in the company will annoy people. In the heat of the moment this can lead to very dangerous comments, assumptions, and truisms which will come back to bite people in the butt a week later. Be aware of this and don't fall into the same trap. Instead be ready to bail these guys out later on. Get the facts, and before you say anything have a backup that can verify what you are saying.

Make your knowledge level clear.

If people ask you what is going on don't say "no comment" as that implies you know something but are not allowed to say it. Simply state that you are not in a position to know yet but that you are investigating.

> **This last point is very important.** Your job is to communicate with everybody in the company. People already assume that you have a much more in-depth knowledge than they have. As developers get very paranoid about changes in the company this can mean that people think you know all about what annoys them but are holding back as you are siding with "the company" or "the management". Don't lose your contact with and the trust of the developers in the company over something you have no control over.

Connecting with your internal developer colleagues

Traveling the world and seemingly spending your work time aimlessly surfing the web and twittering can give a wrong impression of you losing interest in being a developer. Work around that by not losing the connection with the people in your company that build things and listen to what they do.

More importantly, listen very hard when developers feel hindered in delivering their job, and then talk to their management about these problems. Keep these talks anonymous and show the impact these problems have on delivery, employee retention, and the quality of the products your company builds.

Any change for the better you can cause and any improvement in the areas people are concerned about that can be attributed back to you allow you to not be part of the delivery without losing geek points. You achieve this by talking in the right language at the right time to the right people. Developers are always asked to deliver yesterday what is defined in a week's time and never feel the chance to voice their concerns. Make them aware that you can be their spokesperson and that you are there to demand time for discussing problems with their managers.

Working with PR and Marketing

As stated in detail beforehand, your role as a developer evangelist means that you are in between classic outreach departments like PR, Marketing, and Developers. The danger there is that these departments could see you as competition. Therefore, it is immensely important that you keep on good terms and in constant communication with these departments. The reasons for this should be obvious:

You don't want to give mixed messages.
Different views, yes, but you should both at least promote the same products to different audiences.

PR and Marketing know legal implications that you don't know.
Make sure you chat with them before prematurely releasing information or promoting products that are about to drastically change.

These departments have already established communication channels.
This can give you a good and valuable inroad to speak at conferences and work with the press.

You can learn from their experience.
Most probably these departments will have people that are on the job longer than you have been and can predict patterns.

You can feed back state-of-the-art developer information to these departments.
Validating the impression PR has of a new product with realities, makes sure that over-ambitious advertising doesn't promise developers functionality that you'd have to explain is not available yet.

Sharing contacts can open doors.
You may have a way into companies and publication channels that PR always wanted to have but couldn't find.

Only by liaising with the other outreach channels of the company you can make sure that you give the same message. You don't want to be seen as a threat.

Being known as an outward channel

As a lot of people talk about the company to the world on different levels, it is a good plan to tell the company about your outward communication channels. This ensures that you are not approaching the same people in parallel and possibly give mixed messages effectively undermining your and your company's credibility. It also shows the company that you are a person that reaches where they can't. List all the places where you publish:

- Blogs
- Conferences
- Forums
- Magazines (print and online)
- Mailing lists
- Professional groups and bodies
- Social apps
- Social networks

Also make sure to tell people that you have invite codes and accounts for products (if you have them of course). This works a treat and stops people from having to sign up themselves if they just want to have a look.

Training evangelist and developer colleagues

Just like clever developers share as much of their knowledge with other developers, you should plan for training people in the company to do what you do. The reason is the same in both cases: you can share the workload and you can be sick or take a holiday. Another thing you can do is target new ideas and spend time on other plans dealing with your (professional) life.

Training evangelists is a tricky thing. By definition, evangelists should be found and empowered and cannot really be "made". Just like you use your powers of communication and persuasion to bring products to developers you can use them to bring potential evangelists out of the woodwork:

Make the company aware of the communication channels to the outside world.
Say that the blog you have is successful and that you are very happy to publish in-depth blog posts about current work and best practices used in the company. Then offer help with writing those.

Tell the company about events: both the ones you organize and ones that happen in the area.
This is especially necessary when you can't attend them. Offer to support a developer who wants to go with free goodies to give out (stickers, shirts…) - if they go and bring back photos and information how it went to write a blog post together.

Offer specialized internal training and talks.

As the prospect of evangelism might still be alien and even scary to people, cut the training offer down to things that can be applied in any professional role: writing for the web, public speaking tips, finding great web content, etc.

Share great responses from the outside world.

Send out a newsletter of "happy geek quotes" with tweets and blog posts about how a certain event or product launch was received in the developer world.

Ask people to challenge your products

Run some internal competitions to change or collect ideas about how people in the company would like its products to change. Internal company hack days are good for this.

A lot of this will tie in with communication channels that PR and Marketing already use. Ask them for help instead of doing your own thing and creating confusion and trespassing on their territory.

Sharing useful technology

As a developer evangelist, you will have the hand on the pulse of technology. Not everybody has the time to keep up the same way, hardly anybody has. That is why a very interesting part of your job is to communicate great technology finds with your company.

If you find great tools that make everyone's life easier, share them with the company. This can include things like screenshot tools to stop people from sending you Word documents with embedded resized bitmaps, translation tools, communication tools, and basically anything that you use to save time every day.

Balancing your personal and official channels

One issue you will face sooner or later as a developer evangelist is **where** to put your information. The big mistake is to use your own, personal channels (blog, Twitter account, Facebook page and so on) for everything. This is tempting to do, for several reasons:

You control the timing and style of the publication.

It is your blog/twitter account/google+ profile, etc.

It aids your personal brand.

People come to you to find out about interesting, awesome, up-to-date technical info.

It is lucrative.

You can make some extra money by showing ads or getting other sponsorships.

This is how a lot of personal blogs work, and do well at that. The issue, however, is that once you publish for a certain company you are not just "you" any longer. You are a translator and a voice of a certain company or product and, more importantly, the people working in or on it. That's why it is a very bad move to use your own channels as the source for company or product specific information. For various reasons:

You limit yourself.

By posting about a certain company or technology you become the person to cover that technology. You have to answer comments you can not answer but the people working on the technology can. What happens when you leave the company or move on to another product?

It will cause discontent.

While it is good that you use your fame to promote the work of your colleagues, you also get benefits (tangible in the form of advertising money and lesser tangible in the form of fame) by writing about the work of other people. You didn't do the hard work, you didn't attend the meetings, you didn't work extra hours to meet deadlines but you become the face of the project. This can be seen as ripping people off.

You can cause a disconnect.

Your channels are not where the product is built and maintained. What happens when the technology changes? Keeping the information close to the subject matter and maintained by the people who work on the product ensures that things stay up-to-date without your intervention.

You miss out on internal promotion.

An official blog or repository of your company is promoted by the Marketing department and everyone in the company. People know it as a channel and are excited to see new and up-to-date, well produced information there. Your own blog might not be known to people in the company and is not seen as a trustworthy channel because it is not in control of the company. You could go rogue any time, or your blog could be hacked. Official channels will shy away from promoting your work with you.

You cause a massive maintenance overhead.

It is up to you to keep the posts on the subject up-to-date when changes to the product happen. You might not be with the company any longer when changes happen, and you don't want to have to deal with questions by people years later when "your product" all of a sudden changes.

All in all, the balance between publishing on your own channels and company official channels is simple: publish detailed information tied to the product where the product is maintained: a company wiki, an official blog maintained by the company, the GitHub repository and similar places. That way you can move on to another company without leaving confusion and a mess behind. It also ensures that web searches in the future find a maintained resource, and that social media updates don't end up in a missing page: a company has a stake in keeping an official resource available.

As an evangelist, your job is to raise the profile of the company creating the product you promote and the product itself. Thus the official pages with the up-to-date information should show up higher in search results than your own blog. Otherwise you failed at doing your job. You are a pointer, a transmitter, not a replacement.

What you publish on your own channels should whet the appetite of your readers to go to the official place and learn more. This could include translations, a quick screen-cast, screenshots, a nice demo using the technology (instead of being it). Think of your company's product as a movie and you are the one who cuts and releases the trailers in various places to get people excited about it.

Finally, there is a very fine line between promoting your company's work and taking credit for it. Don't cross it and you'll help the company and keep being a trustworthy resource and communication channel.