Meet the Next Code Camp Speaker: You!

Some tips on tech speaking by JeremyBytes.com

Overview

Code Camp speakers come from the developer community. That's you! It turns out that most developers have something to say. We have all picked up tips and tricks with each new project we work on. In addition, we constantly find things in the development platform that we weren't familiar with before. And many times we find ourselves saying, "I wish someone had told me about this before."

Now's your chance. Code Camp is an easy place to get started as a speaker. As a developer, you are speaking to other developers. We all understand each other. Also since Code Camp is free, we're not expecting professional technical speakers (although, I'll admit that I've heard speakers at Code Camp who were better than some of the speakers at the professional conferences).

I'm going to share some tips that I've learned along the way, including some things that I specifically set out to avoid. There are no rules for speaking. Everyone has a different style. These are a set of tips which have worked for me. They may or may not work for you.

Danger: You May Get Hooked

I had done a few presentations for fellow developers within my department at work but never thought I would get up to speak in front of complete strangers. But after speaking at my first Code Camp, I was hooked. I signed up to speak at more Code Camps in my area, and let the local user groups know that I was interested in speaking.

Here's what hooked me: there's nothing quite like helping other people. The speaking part itself is a lot of fun for me, but what is most rewarding is if I get an email a week later from someone who let me know how they were able to use the information that I presented. Being helpful to someone else really drives me forward.

How Did I Get Started?

I got started as a speaker almost by accident. In my mind, I didn't have anything ground-breaking to talk about. I wasn't dealing with the latest beta products or cutting-edge software. I was just a competent developer working with established technologies. It seems like everyone was interested in whatever was new and shiny, so I didn't think I had anything to say.

Then I was listening to .NET Rocks! (a really good podcast, by the way). There was a panel discussion and one of the topics that came up was the lack of beginning and intermediate sessions for people who were new to development or just new to .NET. When .NET was first released, there were all sorts of introductory sessions including the basics of the languages, the CLR, garbage collection, and just how to get started creating applications. Then over the years, the introductory sessions faded away to be replaced by whatever the latest technologies were. The panel agreed that there was a need for people to present those introductory topics to keep bringing more people into the world of .NET development.

I thought, "Hey, I can do that." And I took it as a challenge. About 2 weeks after the podcast, I heard about an upcoming Code Camp and determined that I would put together three sessions for it. (It may have been a lot to bite off my first time around, but I had 8 weeks to prepare).

Those first sessions went very well. And I attribute that primarily to the preparation.

So let's take a look at some tips to get started with.

One Rule

Earlier I mentioned that there were no rules. But that wasn't quite true. In reality there is one rule: Have fun! If you aren't going to enjoy this, then no one attending will enjoy it either. It's okay if you are nervous. But once you get started, just like any other performance, the nervousness usually fades away. Just go with it and have fun.

Picking a Topic

One of the first things you will need to do is pick a topic. Pick a topic that you are passionate about, something you really love. Okay, love is a bit of a strong word, but select something that you have found really useful in your development -- something that you wouldn't want anyone to take away from you.

Here's a tip from Scott Hanselman: Give the presentation that you want to attend. When you are thinking about a topic, go back to the "I wish someone had told me about this before" moment. It may be a common subject; it may seem basic or simplistic. That's okay. If it is something that you wish you had known about earlier, chances are that there are a lot of other developers who don't know about it yet.

One other thing about picking a topic: be aware of the scope. You have limited time to present your session. If you pick a topic that is too broad, then you may end up skimming over the top without going into any useful detail. If you pick a topic that is too narrow, you may find that you don't have material to fill up the time slot.

Preparation

I cannot say enough about being prepared for your presentation. I know what I am going to say before I walk in the room. If I end up giving the same presentation a second or third time, it will generally be almost exactly the same content each time.

Often at Code Camp, you will see people in the back of the room frantically typing away. These are usually speakers who are finishing up their slides or code samples. This works for some people (people who love the "deadline"). And if you are a veteran speaker, you can often get away with it. It doesn't work for me. Your first time out, you are going to have a lot of things on your mind (How do I use the AV equipment? Is Visual Studio working right? Can I find my slides?) If you have your presentation nailed down, that is one less thing to worry about.

The first step in preparation is to know your material. By putting together a presentation, I find that I have big gaps in my knowledge about the topic. This is usually because I needed to use a technology for a specific project with specific requirements. But if you want to present the broader topic, you will find that there are parts of the technology that you have not used and need to learn about. This doesn't mean you need to become an expert. But you do need to at least know all of the terminology and have a general idea of how ti all works, even if you can't give the specific details.

Once you have everything put together (topic, slides, demo code, etc.), then practice, practice, practice. Run through the presentation all the way through multiple times. And not just in your head. Speak out loud (pets are a great audience for your first practice runs). I generally run through a new presentation at least 4 times. This helps me find the places where I stumble or need to clarify my thoughts. Sometimes you get tongue-tied. That's what practice is for. Even if I am doing a repeat of a presentation that I've done before, I'll run through 1 or 2 practices to make sure that I still know what I'm talking about.

Stay on Time

One thing that all of the practice will help you with is to stay on time. Is your presentation running long? Figure out what you can cut or speed up. Is your presentation short? That may be okay. It leaves more time for questions or to explore some of the items a little further.

One of the worst things you can do at Code Camp is run long. Code Camp is usually a very compressed day. People have limited time to get from session to session (and to visit the restroom or grab a soda in between). If you run long, then you will lose the attention of the people in the room anyway, so it's best to make sure that won't happen.

Things don't always go according to plan. My first time out, I thought that I had my timing down fine. I was confident and took time for a few questions in the middle of the session. But because I had practiced so much, I recognized about halfway through the time that I wasn't halfway through my material. I made some quick mental cuts and decided what I could skip over. I was able to do this on-the-fly without it being too obvious that I had done so.

Slides

No one likes slides at Code Camp. We're not executives that want bullet points. We want to see code in action. So feel free to go light on the slides.

With that said, I would not recommend ditching the slides entirely. If you have slides, then the folks in the room know that you spent at least a little bit of time in preparation. I've gone to a couple completely slide-free presentations, and they seemed to be completely ad-libbed (which could be fine depending on the topic). My personal approach is to have a couple of slides that introduce me and the topic, but then I spend most of my time in the code. This is just my style; other people approach this differently. Again, the goal is to find what works best for you as a speaker.

Remember the advice to give the presentation you would like to attend. Do you want to see a lot of wordy slides with the presenter simply reading them? Probably not. The slides are there to augment what you have to say, not repeat it or replace it.

One other thing that I do with my slides is have a reference links slide at the end of the presentation. This would contain links to web sites, blog articles, MSDN articles, or other helpful materials. I post my slides on my website so that people can get to these links without having to scribble them all down during the presentation.

Demo

Code Campers love demos. We want to see what you are talking about in action. There are a lot of different styles for this part.

First, there is the advice to never do a live demo. There are too many things to go wrong. And Murphy's Law says that it will probably all go wrong right in the middle of your presentation. Personally, I think that preparation can mitigate a lot of this risk (and I'll talk about a few specific items of preparation that help in just a bit).

If you have a demo set up that is very complex, such as one that requires multiple computers networked together or needs access to a specific web site, then you might consider doing a screen capture of the demo and then playing it during the presentation. And even if you want to do this live, it doesn't hurt to have the video as a backup in case the network goes down or something else goes wrong.

But the most effective demos are live demos. How you approach this will depend on your topic and how deeply you want to cover things. You may want to start with a blank project and build an entire application live. This works well in a lot of cases, but be aware that you will be limited in how much you can show.

You may want to start with several different projects that are in various stages of completion. In this case, you can open the #1 project, add some code and show what it does. Then move on to the #2 project which has more code in place, add some more code and show what that does. This can be a very effective way of presenting a topic.

Here's what I do -- again, take this as part of my personal style based on how I pick topics; it is what has worked well for me. I generally start with a project that is prepared just enough for me to start putting in the important code. For example, if I am doing an introduction to XAML, I start out with a completely new, blank project and move from there. If I am speaking about lambda expressions, then I have the UI already laid out and start with mostly blank code behind the UI. The idea is that anything that is already in place is not necessary to understand for the topic being discussed; it is part of the background.

One other thing you can do. If you have large blocks of code to add as part of your demo, don't make people watch you type. This can be especially painful if you are not a good typist. If you have large blocks of code, then create another file within your project that contains the code in comment blocks. Then you can copy/paste the blocks of code into your project during the demo. This saves you a lot of time, and it also helps to avoid typos.

As part of my preparation, I have 2 copies of each of my projects. The first copy is the "Starter" project. This is the mostly-blank project that I am going to fill in with the code during the presentation. The second copy is the "Completed" project. This is what the project looks like after I have completely run through the demo. This is my safety net. If something goes horribly wrong during the live code demo, I can always open up the completed application and show the pieces already in place. The good news is that I have only had to do this once (knock on wood). I attribute this to preparation; I have practiced the code demo so many times that I know exactly what I need to do. And if something does go wrong, I've probably already seen that same issue during the practice runs.

The Tools

Feel free to ignore this tip. This is a personal preference based on sessions that I have attended. The tip: run naked. Okay, don't do that literally (no one needs to see that). What I mean is disable the add-ins in your development environment unless they are critical to what you are trying to demo. When I see someone do a demo, I want to be able to go home and do exactly the same thing. If the speaker is using an add-in that has magic keystrokes, then it makes it harder for me to follow.

This primarily applies to non-free add-ins. If you have free tools that you find useful, then by all means recommend them. As developers we are always looking for new tools that can make our development faster and more effective.

One other thing about tools: if you know any shortcuts that are native to the development environment, be sure to share them. Not everyone is aware of what Ctrl+K,D does. Not everyone is aware of what the little purple box means. Not everyone is aware that you can press 2 keys to stub out an interface. If you can fit these tips into your presentation, then go ahead and do it.

Organization

How you organize things is part of your preparation. You need to know where all of your slides, code samples, demos, and websites are. I have been in a few presentations where the speaker opened the wrong projects and had trouble finding the code he was looking for. Don't let this happen to you. Something that I find useful is to create a folder that has shortcuts to everything I need. You can put this folder on the desktop or even dock it to the taskbar. If you have multiple shortcuts to code projects, then number the shortcuts so that you can open them in the correct order.

During the Presentation

You will probably be jittery as the presentation starts. That's fine. This is normal. A lot of times, this may lead to you speaking too quickly or getting tongue-tied. If this happens, just pause and take a deep breath. Have a bottle of water with you, and take a sip. You don't have to fill every second with talking. Taking a deep breath will give your brain a chance to catch up and organize your thoughts. Then you can continue clearly and confidently.

Watch your speed. If you have practiced, then you should know where the quarter, half-way, and three-quarter marks are in your presentation. When you hit those spots, how does that compare to the clock? If you are ahead of time, then you may want to slow down, or pause to see if anyone has questions. If you are running late, then you may want to start thinking about what you can skip later on or if you can simply speed up a little bit.

Focus on Individuals

When you are speaking, try to look at individuals in the room. It is tempting to talk directly to one person (the one person who is paying most attention). But make sure that you are talking to everyone. Talking to "everyone" does not mean simply throwing your words into the room; you need to talk to the individual people in the room. While speaking, take a few moments to make eye contact with one person, then move on to another person. Remember that you are talking to actual people.

I forgot this once. I was on my third presentation of the day. The other two presentations had full rooms with lots of energy. The third presentation just had a few people in the room. I tried to continue the energy into that room, but it didn't work. It was awkward, and I bombed the presentation. I got all of the information out there (because of the practice I had), but I never made a connection with anyone in the room. Ultimately, I'm glad that it happened. I learned from what I did wrong, and I've been better able to adjust my presentation style based on the situation.

Taking Questions

It's up to you to decide whether you want to take questions during your presentation. If you are going to take questions freely, then plan your presentation so that it would normally end 10 to 15 minutes before the end of the allotted time. If you want to save questions for the end (if there is time), that is fine, too. Just let your attendees know how you would like to handle questions.

If you take questions, stay on topic. If someone asks a question that will take you off topic, it is okay to put the question off. As developers, we always want to show off how much we know. So if someone asks a question, your first instinct is to answer it. But here's the problem: if the question takes the session off topic, it may be good for the one person in the room (the person who asked the question), but it will be horrible for the rest of the people in the room. After all, they all came to hear you speak about a particular subject. I have often been frustrated (as an attendee) when a session was derailed by someone asking questions that are relevant only to that one person.

If you take questions, it's okay to defer. It may be that the question is either off topic or that the question is on-topic but would take too long to answer. If this is the case, then let the person know that you would be happy to talk to them about it later, either between sessions or in email.

If you take questions, it's okay to not know the answer. You don't have to know everything before you become a speaker. I have often had people ask me a question that I did not know the answer to. This is usually because they bring up a scenario that I had never thought through or tried before. In these situations, I generally write down the question and then say that I will post the answer on my blog once I have time to research it. Exactly how you handle the situation is up to you. But it is better to say that you don't know than to make up an answer.

After the Presentation

There are a couple things that you can do after the presentation to wrap things up. First, if you have code samples or slides that you say you will make available for download, then do it. I have been at professional conferences and had the speaker say that the code from the session would be on the website after a week. I checked after a week, and it wasn't there. I checked after two weeks, and it still wasn't there. Because of curiosity, I went back two months later, and it still wasn't there. If you say that you are going to post your code, then post your code. And by the way, if you don't want to post your code, that's okay, too. Just make sure that if you say you will do something, you follow up on it.

Another "this works for me": I always post my code samples and slides on my website before the presentation starts. This way, I don't have to worry about finding the time to do this later (because I'm very good at forgetting things). It also has the side-effect of forcing me to prepare. I can't post the slides and code samples until after my preparation is complete.

Finally, follow up on questions. If you say that you will post answers on your blog or follow up with an individual over email, then make sure you do that.

Final Words

So, we've talked about a lot of different tips. As I noted, most of these things have worked well for me. Some of them are things that have bothered me from sessions that I have attended. But everyone is different.

Remember the one rule: Have fun! The fate of the world does not depend on you giving a good presentation. If you mess up, no one's going to die. You learn best when things go wrong.

And make sure to be yourself. Everyone has a different style of speaking. Everyone has a different style of coding. Don't pretend to be someone else when you are speaking. Just be yourself. If you are prepared, then things are bound to turn out just fine.

Wrap Up

Everyone has something to share with other developers. Take the next step and sign up for the next Code Camp in your area. You might find that you love it, and you'll be speaking more and more. You might find that it's not your thing. That's okay, too. But you never know until you try.

Happy speaking!