ACCESSIBLE CONTINUOUS INTEGRATION

Stanford
Drupal Camp

CivicActions

- → Nerdstein
- → Continuous Integration
- → Current Limitations
- → A Future Vision
- → Accessible Continuous Integration
- → Case Studies
- → A Call to Action

Nerdstein (Adam)

- → Associate Director of Engineering, CivicActions
- → Masters of Science, Information Systems Security
- → Drupal 8 Maintainer of Taxonomy Menu, Password Policy, Key, Encrypt, Field Encrypt

Before happy hour…

Let’s do some
role playing!

Kenneth Parcell

1. Intern in IT Department
2. Hard-working, eager, team player, well loved, still learning IT
3. Screwed up last three code releases
4. Likes knitting and kite flying

Ron Swanson

1. Runs IT Department
2. Non-technical
3. Low budget
4. Loves scotch and eats meat, actually hates IT

Cruella Deville

1. IT Client
2. Really non-technical
3. True honey badger, doesn’t give a ….
4. Eats babies as a hobby

Continuous Integration

CivicActions

The Slack conversations are getting ugly….

- ➔ **Cruella:** *Your next Drupal release better work or I'll eat your babies*
- ➔ **Ron:** *Team, go figure this out. Mmm… scotch*
- ➔ **Kenneth:** *Now's my chance! I will make this right!*

It’s life or death

Kenneth stumbles on a concept called Continuous Integration...

→ **He found an awesome quote to motivate:**

“Don’t solve the same problem more than once”

→ **Kenneth looks up deployments on Drupal.org:**

Use Drush to script deployments

→ **He finds a bash script to run on servers!**

Kenneth presents this news to Ron...

- **Ron responds, like a boss:**

  "I was close to firing you, so make sure other engineers can do this. Oh, and pour me more scotch."

- **Kenneth searches for collaborative Continuous Integration tools:**

  Use Jenkins for distributed builds and synchronize continuous integration across teams

- **He used Jenkins to integrate the bash scripts with their servers and gives the other engineers access to Jenkins**

DevOps automates solutions to longstanding CI problems

ACCESSIBLE CONTINUOUS INTEGRATION | CONTINUOUS INTEGRATION

- ➔ **Continuous learning into applied problem solving**
- ➔ **Consistency equates to predictability and stability**
- ➔ **Automation over error-prone manual processes**
- ➔ **Have no barriers:** *release management, security scanning, log analysis, 508 compliance, automated testing, quality assurance, code reviews, on-demand environments*

You offer a better service to your users with CI practices

Current Limitations

Kenneth runs the next deployment in Jenkins successfully...

➔ **Kenneth emails Ron and Cruella:**

“Your deployment is complete! Go take a look, it actually works!”

➔ **Cruella responds:**

“I can’t believe we’re paying you to do this, why can’t I run deployments?”

➔ **Ron responds:**

“Finally, freedom. Kenneth, let Cruella have access to Jenkins.”

Kenneth provides Cruella access to Jenkins...

➔ **Cruella escalates to the entire organization:**
“What kind of insane technical system is this? There is no way I’m using this, do you think I’m Steve Jobs?!”

➔ **Ron adds to the list of complaints he receives across the board**

- ◆ *Your security scan results are too long and illegible*
- ◆ *I can’t effectively test before my code is deployed*
- ◆ *I have thousands of unread emails, why do you keep emailing me?*

ACCESSIBLE CONTINUOUS INTEGRATION | CURRENT LIMITATIONS

Innovation in technical automation is not enough…

→ Why shouldn’t Cruella have access?

→ DevOps has only emphasized technical problems for technical audiences marketed as a better service to those we serve

→ It’s time we evolve Continuous Integration beyond just a technical practice

We’re missing those we serve in the process

A Future Vision

Innovation is not just technical breakthroughs

- → The problem is not the tools, it’s that they are not approachable
- → Open a dialogue with those you serve

Build both technical and social bridges

Digital Enablement

Let’s promote digital enablement for Continuous Integration…

- ➔ Emphasize ease of adoption and removal of barriers to entry
- ➔ Promote effective information sharing and transparency
- ➔ Drive toward usability of your services and tools
- ➔ Deliver comprehensive and streamlined services

How can we frame high level goals…

→ **SIMPLE - Processes void of encumbrance**

→ **USEFUL - Solve meaningful problems**

→ **FLEXIBLE - Build robust, long-term, unassuming solutions**

→ **TRANSPARENT - Communicate concisely and frequently**

We aim to take CI to the masses

Accessible Continuous Integration

Wikipedia defines accessibility as…

> *The process of creating products that are usable by people with the widest possible range of abilities, operating within the widest possible range of situations.*

Do not get confused with

508 Compliance Accessibility

There are current practices that build bridges…

- ➔ Abstracting technical details
- ➔ Systems integration
- ➔ Streamlined processes
- ➔ Platform and device agnostic

1. **Abstracting technical details**
2. Integrate systems
3. End-user involvement
4. Platform and device agnostic

Abstracting technical details...

- → KISS concept (Keep it simple, stupid)
- → Systems must promote usability, account for technical literacy
- → Build finely tuned user interfaces, not command lines
- → Limit decision points, add help text, consistent UI design

1. Abstracting technical details
2. **Integrate systems**
3. End-user involvement
4. Platform and device agnostic

Integrate systems...

- ➔ Select systems that promote interoperability (future proof for continuous learning)
- ➔ Connect systems instead of forcing users to use multiple systems
- ➔ Make use of web services, APIs, plugin systems, and give back to communities so others can benefit

1. Abstracting technical details
2. Integrate systems
3. **End-user involvement**
4. Platform and device agnostic

End-user involvement...

- ➔ Identify systems in which your users are comfortable using -- enhance them
- ➔ Avoid forcing users to learn too much or use new systems
- ➔ Encourage user testing and feedback loops to participate in Continuous Integration discussions

1. Abstracting technical details
2. Integrate systems
3. End-user involvement
4. **Platform and device agnostic**

Platform and device agnostic…

- ➔ Users want access everywhere and immediately
- ➔ Avoid systems (like email) where communication can break down
- ➔ Adopt best of breed solutions that don’t restrict platforms
- ➔ Systems should not only be on desktops, use of mobile phones, tablets, and refrigerators (Internet of Things)

Tear down the walls of CI participation

Case Studies

CivicActions

Kenneth read that CivicActions integrated Slack and Jenkins...

- → **He built a command in Slack to perform releases**

  Including instructions and commands by environment

- → **Kenneth trained Cruella to run deploy commands**

  She already ran Giphy commands in Slack and was comfortable participating

Cruella’s heart grew three sizes that day…

- → Kenneth got a job offer after a recommendation from Cruella
- → Cruella took Ron out for scotch

A true partnership was formed

ACCESSIBLE CONTINUOUS INTEGRATION | CASE STUDIES

Accessible CI in the wild...

1. Slack and Jenkins
2. On-demand environments
3. JIRA and BDD

Slack and Jenkins is a happy marriage...

- → **Slack is highly intuitive for non-technical users**
- → **Slack supports custom commands with help text**
- → **Commands can include running Jenkins commands**
  - ◆ *Abstract parameters and options into separate commands*
  - ◆ *Customize output to Slack so it’s relevant to all users*

Accessible CI in the wild...

1. Slack and Jenkins
2. On-demand environments
3. JIRA and BDD

On-demand environments improve QA and Client Sign-off...

- ➔ Helps project managers and product owner perform testing
- ➔ Leverage cloud hosting to produce branch-level environments
- ➔ Integrate as a Slack command, produce URL and credentials
- ➔ Simplifies release management; code is not merged into a release until a branch has been signed off

Accessible CI in the wild...

1. Slack and Jenkins
2. On-demand environments
3. **JIRA, BDD and Code Repositories**

BDD can be improved by using JIRA and Repo Hooks...

- ➔ Behave for JIRA plugin, integrates with code repository
- ➔ Empower users to write automated tests within JIRA tickets
- ➔ Leverage hooks in the repository to run automated tests when developers submit pull requests (TravisCI or Jenkins)
- ➔ Feedback loops are drastically shortened between user needs and developer's code

A Call to Action

CivicActions

Unlock the potential

Empower your users

Accessible Continuous Integration

Thank you, Stanford!

Questions?