

The Project Application Process, Revisited

Greg Dunlap
Alan Palazzo
Angela Byron

The process

1. Create a sandbox
2. Commit your code
3. Create an issue in the “Project applications” queue
4. Wait for someone to review/RTBC it
5. Profit!

The problem

1. Create a sandbox
2. Commit your code
3. Create an issue in the “Project applications” queue
4. Wait for someone to review/RTBC it
5. Profit!

Step 4 tends to turn this...



I love Drupal!! I can't wait to give back!

...into this. :(



Screw you guys. I'm going to Github.

Why do we do this to people?

- Impart community knowledge (coding standards, best practices, etc.)
- Prevent proliferation of insecure modules
- Prevent module duplication
- Reduce insecure/broken code
- Ensure license/policy compliance

So, is it effective?

Here's what the data shows.

What data we gathered

- Spot-checked ~60 applications (mix of approved/declined), checked for:
 - Reasons applications were sent back
 - What happened after approval/denial
 - Number of days people were in process

<http://lb.cm/project-application-stats-spreadsheet>

Reasons for “needs work”

Rank	Reason	Percentage
1	Coding standards	64%
2	API usage	45%
3	Application rules	33%
4	Duplication	19%
5	Legal or external libs policy	12%
6	Security	5%

Conclusions

- New developers don't know coding standards, nor have in-depth knowledge of Drupal APIs yet.
- Duh; neither did you when you were new.
- Our application rules and licensing policies are confusing.
- It's hard to find modules on drupal.org.
- (generally) Only security team members find security holes in new modules.

Process sustainability

Queue Size	Currently, there are <i>423</i> active apps in the queue. <i>202</i> of these are sitting in 'needs review' status.
Application Age	Oldest CNW application: <i>57 weeks</i> Oldest 'needs review' application: <i>49 weeks</i> Oldest application with 0 comments: <i>9 weeks</i>
Apps In vs. Apps Out	In the last week: <i>21</i> New Application <i>9</i> RTBC/Fixed Applications
#of active reviewers	<i>Varies, but estimated between 5 and 10 for any given week.</i>

<http://jthorson.doesdrupal.com/project-apps-pt1>

Average length in queue: 88 days

Conclusions

- Process is unsustainable: too many eager users, not enough people helping
- However, we do get a number of benefits:
 - Easy way to impart Drupal community norms on new people
 - Easy way to catch legal issues before they happen

**So what do we do
now?**

#1: Figure out our priorities

What behaviour do we want to promote, what
behaviour do we not want to promote?

#2: Focus on automation

Keep humans on things humans do well; let machines handle coding standards/security/legal review.

#3: Separate *mentorship* from *access*

Create a view of new peoples' commits. Have code review team focus on helping those people.

#4: Create better metrics/search tools on drupal.org

Don't take the lack of these tools out on eager new
people.

Concrete proposal

- Get jthorson's automated Coder review code deployed on d.o
- Expand with Legal / API sanity / security checking
- Display Coder status on project page to indicate project quality to maintainers on full projects and all users on sandboxes
- Feed data into Solr to make search not suck
- Add "app review" bingo
- Add steps for what new reviewers can do
- Add git clone command to project issue

Other ideas

- Move reviews to first stable release, rather than first submission
- Enable dev releases on sandboxes
- Grant full project upgrade only to projects with stable releases
- Time-box ability to get a namespace (e.g. 2 months since first push)

**But seriously, let's figure
this out this time.**

The real proposal

- Automated coder review on project page
 - Sec. / legal to follow
- Allow dev tarballs on sandboxes
- Move approval process to stable release, limit project namespace to stable release