Enabling Data-Driven API Design with Community Usage Data: A Need-Finding Study

Tianyi Zhang\textsuperscript{1}, Björn Hartmann\textsuperscript{2}, Miryung Kim\textsuperscript{3}, Elena Glassman\textsuperscript{1}

\textsuperscript{1}Harvard University \quad \textsuperscript{2}UC Berkeley \quad \textsuperscript{3}UCLA
APIs are ubiquitous

Web APIs
- Google APIs
- Twitter
- Instagram
- Facebook
- AWS
- Yahoo Finance
- Vimeo
- Stripe

SDKs
- Java
- JavaScript
- Python
- Julia
- Scala
- Microsoft .NET

Frameworks & libraries
- TensorFlow
- jQuery
- PyTorch
- NumPy
- Apache Spark
- Spring
APIs are a primary interface between programmers and computers
What Makes APIs Hard to Learn? Answers from Developers

Martin P. Robillard, McGill University

Most software projects reuse components exposed through APIs. In fact, current-day software development technologies are becoming inseparable from the large APIs they provide. To name two prominent examples, both the Java Software Development Kit and the .NET framework ship with APIs comprising thousands of classes supporting tasks that range from reading files to managing complex process workflows.

An API is the interface to implemented functionality that developers can access to perform various tasks. APIs support code reuse, provide high-level abstractions that facilitate programming tasks, and help unify the programming ex- and interviewing developers about the obstacles they faced learning APIs, I discovered many issues that complement those mentioned in API design textbooks and articles. In particular, I found that API learning resources are critically impor-
Human-Centered API Design

• Apply HCI methods to API design, e.g., A/B testing [1,2,3]

• But it is costly.
  • Too many APIs and usage scenarios
  • Participant recruitment

1. Ellis et al. The factory pattern in API design: A usability evaluation. ICSE 2007
A lot of API usage data has been generated from programmer communities...
How can we leverage community data to inform better API design?
Contributions

• Semi-structured interviews with 23 API designers

• An in-depth analysis of their design styles, usability evaluation methods in practice, and unmet information needs

• Several tool design implications for leveraging community API usage data to inform API design
Interview with 23 API Designers

- Large Tech Companies: 11
- Academia: 10
- Open-source: 2

- Web APIs: 11
- DSLs: 6
- Libraries: 6

Software Logos:
- Caffe
- MxNet
- Julia
- Vega-Lite
- Z3
- Chisel
- Gen.jl
- Alloy
- Partisan

OpenSci
Interview Questions

• What design decisions and tradeoffs have you made?

• How do you evaluate the usability of your APIs?

• What challenges and frustrations do you have?

• What information would you like to discover?

• What tool support do you need?
Finding 1. Designer Spectrum

User-driven  Self-driven  Visionary  Closed-world
User-Driven API Designer

- User-centered design process
- Survey user needs
- API stability
Self-Driven API Designer

• Make their own decisions
• Clear goals and priorities
• API expressiveness & extensibility
Visionary API Designer

- Lack communication channels to users
- Design with imagined use cases
- Eager to know API usage in the wild
Closed-World API Designer

• APIs used internally in a company

• Work closely with all stakeholders

• Least tension between API designers and users
Finding 1. Designer Spectrum

User-driven  Self-driven  Visionary  Closed-world

They all acknowledged the importance of keeping users in mind.
Finding 2. Usability Evaluation Methods

- Reading bug reports, emails, online discussions: 18
- Peer review: 13
- Regression testing on existing usage: 6
- Monitoring and logging web API traffic: 6
- Teaching in classrooms and workshops: 5
- Building example galleries: 5
- A/B testing: 4
- Scheduling regular meetings with users: 4
- Sending out surveys: 1
- Cognitive dimensions: 1
Finding 3. Unmet Information Needs

<table>
<thead>
<tr>
<th>Information Need</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>A holistic view of real use cases</td>
<td>15</td>
</tr>
<tr>
<td>A rich description about user’s mental models</td>
<td>12</td>
</tr>
<tr>
<td>Common mistakes and workarounds</td>
<td>11</td>
</tr>
<tr>
<td>API call frequency</td>
<td>9</td>
</tr>
<tr>
<td>Behavioral metrics beyond static usage</td>
<td>9</td>
</tr>
<tr>
<td>Backward compatibility</td>
<td>7</td>
</tr>
<tr>
<td>Comparing and assessing similar libraries</td>
<td>7</td>
</tr>
<tr>
<td>Design inconsistencies</td>
<td>6</td>
</tr>
</tbody>
</table>
Finding 3. Unmet Information Needs

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A holistic view of real use cases</td>
<td>15</td>
</tr>
<tr>
<td>A rich description about user’s mental models</td>
<td>12</td>
</tr>
<tr>
<td>Common mistakes and workarounds</td>
<td>11</td>
</tr>
<tr>
<td>API call frequency</td>
<td>6</td>
</tr>
</tbody>
</table>

P13: “I’d like to **look at their code** and see if they write code **in the ideal way we want them to write**. Because every API has its own purpose when they’re being designed, and we have some use cases in mind. If not, we’ll probably think if this API is designed in a proper way or we should **create some more obvious APIs to deal with this case**.”
Finding 3. Unmet Information Needs

- A holistic view of real use cases: 15
- A rich description about user’s mental models: 12
- Common mistakes and workarounds: 11
- API call frequency: 9
- Behavioral metrics beyond static usage: 9
- Backward compatibility: 7
- Comparing and assessing similar libraries: 7
- Design inconsistencies: 6
Finding 3. Unmet Information Needs

<table>
<thead>
<tr>
<th>Information Need</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>A holistic view of real use cases</td>
<td>15</td>
</tr>
<tr>
<td>A rich description about user’s mental models</td>
<td>12</td>
</tr>
<tr>
<td>Common mistakes and workarounds</td>
<td>11</td>
</tr>
<tr>
<td>API call frequency</td>
<td>8</td>
</tr>
</tbody>
</table>

P6: “I want to get what people are thinking, **not just whether they are able to use this API**, like that’s what you would get from like mining GitHub repos. But what you are not getting is like this person still **spent three hours googling** how to use this API, and he **spent an hour on Stack Overflow** trying to figure out what was going on, in the end, he got that, but it was a **frustrating attempt**.”
Finding 3. Unmet Information Needs

- A holistic view of real use cases: 15
- A rich description about user’s mental models: 12
- Common mistakes and workarounds: 11
- API call frequency: 9
- Behavioral metrics beyond static usage: 9
- Backward compatibility: 7
- Comparing and assessing similar libraries: 7
- Design inconsistencies: 6
Finding 3. Unmet Information Needs

- A holistic view of real use cases: 15
- A rich description about user’s mental models: 12
- Common mistakes and workarounds: 11
- API call frequency: 9
- Behavioral metrics beyond static usage: 9
- Backward compatibility: 7
- Comparing and assessing similar libraries: 7
- Design inconsistencies: 6
Finding 4. Opportunities for Tool Support

- Mining and visualizing API usage mistakes
- Interactively analyzing population-level API usage
- Adapting elicitation mechanisms to understand API users’ mental models
- Exploring the design space of similar APIs
- Live API documentation
Mining and Visualizing Documented Mistakes

tf.keras.metrics.MeanIoU API is practically unusable without a threshold #39173

dd192 opened this issue 9 days ago - 3 comments

[TF 2.0] tf.assert_equal([], [1.0]) doesn't raise error #32

David-Mao opened this issue on Aug 29, 2019 - 14 comments

David-Mao commented on Aug 29, 2019

Does keras.tokenizer.text_to_sequence simply translate into number vectors, or something more?

Yes  Asked today  Active today  Viewed 14 times

OS P  Darwin
Mobi N/A
Tens-
binar

Experiment, learn, and build with our top services, free for 12 months

Try Azure free >
Capturing **Undocumented** Learning Barriers

- Participants suspect a lot of issues were not reported at all.

P9: “I think the people that use the API with no prior experience perhaps hit the wall quickly. That’s the real issue. But for the most part, I suspect that I missed most of the easy problems going on there because they don’t convey that first barrier with their remote communication.”
Capturing **Undocumented** Learning Barriers

- Compilation/runtime errors are good indicators of undocumented learning barriers.
Conclusion

• API designers desire to have a holistic view of real API use cases.

• Such real use cases are gathered in an informal way due to a lack of tool support.

• New interactive systems are needed to distill a variety of API usage insights from community data.