APIs

What They Are and How to Use Them

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Who We Are

- **Jordan Emmendorfer**, Software Developer
  - 2+ years at the Credit Union
  - Maintains API for Authentication of 3rd party chat system
- **Austin Drouare**, Senior Software Developer
  - 9 years at the Credit Union
  - Experience consuming APIs of 3rd party vendors
  - Maintaining and consuming internal APIs for Computerline and Mobile banking applications.
- **Thanks to Ryan Ballard, Jenny Schiebner, Aaron Parks for contributing content to this presentation**

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY Credit Union

Agenda

- API definition
- Common API formats
- API consumption demo
- Considerations around using APIs
- Overview of API documentation

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

What is an API?

An **Application Programming Interface (API)** is a set of protocols used by programmers to create applications for a specific operating system or to interface between the different modules of an application. (*dictionary.com*)

- Operating system
- Web-based system
- Database system
- Computer hardware
- Software library

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

What is an API? (cont.)

Simplified: An API is how our code can leverage code written by other developers.

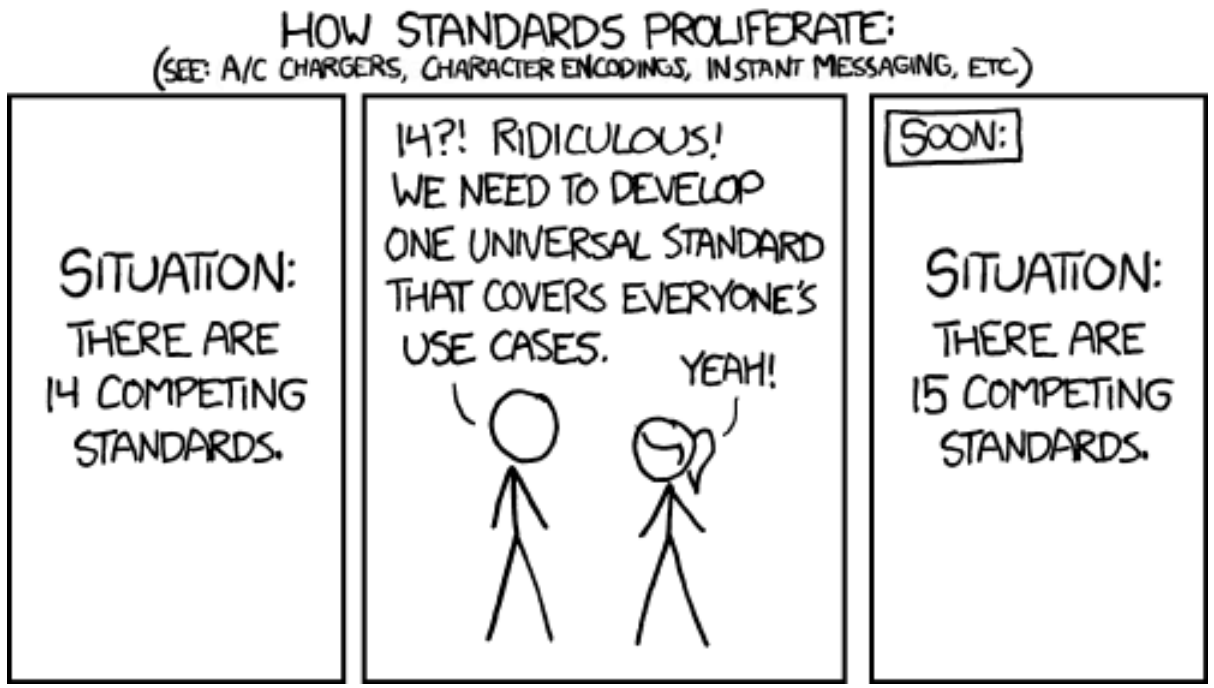
- *Software library*
  The library's API is how your IDE (e.g. Eclipse, NetBeans, IntelliJ) is able to provide autocomplete suggestions.
- *Web-based system*
  Availability: open (public), partner (licensed), or internal (private)
  We will focus on open APIs today.

  Not built in like a library – so how do we communicate?

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

It depends!



https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

API Standards and Protocols

Define how we communicate with APIs

- Remote Procedure Call (RPC) and Remote Method Invocation (RMI)
- Simple Object Access Protocol (SOAP)
- Representational State Transfer (REST) (kind of)
  - REST is an architectural style, not a protocol, but REST implementations typically use a common set of protocols

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

RPC and RMI

- RPC was first, and RMI is the object-oriented analog
- Usually includes tooling to generate stubs and data structures as well as libraries to serialize requests and de-serialize responses.
- First popular implementation was Sun's RPC, best known for its use in Network File System (NFS).
  - Java RMI, DCOM, CORBA, and WCF have also enjoyed popularity in their niches.

https://github.com/oucreditunion/grizzhacks2019

Simple Object Access Protocol (SOAP)

- Descended from XML-RPC
- Formalized by the World Wide Web Consortium (W3C)
- Currently (somewhat) popular, though it is rapidly being replaced by REST and REST-like services.
- Requests and responses are serialized as XML and typically sent by HTTP, though other communication protocols may also be used.
- Contract typically defined in machine-readable format (WSDL and XSD) to support tooling.

https://github.com/oucreditunion/grizzhacks2019

SOAP Request Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:web="http://services.aonaware.com/webservices/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:Define>
         <web:word>test</web:word>
      </web:Define>
   </soapenv:Body>
</soapenv:Envelope>
```

- When dealing with SOAP, language aside, SoapUI is your friend

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Representational State Transfer (REST)

- More general than RPC
  - Can be used to implement RPC or RMI
  - More often presents as a CRUD interface
  - Good APIs are typically a blend
- Works with HTTP rather than against it – makes semantic use of HTTP operations (GET, POST, PUT, DELETE, etc.) and response codes, caching, and so on.
- JSON is the most common serialization. XML was popular too.
- Hypermedia: good APIs include links to other resources in their responses rather than assuming the client knows where to find them.

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

HTTP Overview (for REST purposes)

Headers

- Content-Type
- Accepts
- Authentication
- Explicit Caching

Status Codes

- 1xx – Informational
- 2xx – All good!
- 3xx – Redirect
- 4xx – You messed up
- 5xx – I messed up

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY Credit Union

SOAP vs REST

SOAP

- XML
- Typically HTTP
- More explicitly defined
- More common in internal/private web services

REST

- Typically JSON
- Exclusively HTTP
- More implicitly understandable
- More common in external/public web services

https://github.com/oucreditunion/grizzhacks2019

Live Demo

- https://github.com/oucreditunion/grizzhacks2019
- Consuming a REST API
- If you're still not sure where to start, check out this tutorial: https://www.restapitutorial.com/

Should I use an external API?

Pros

- Leverage work that's already done
- Maintenance is outsourced
- Focus on the core functionality of your project

Cons

- Additional overhead/latency
- Usage limits
- Risk of API going away or changing without notice
- Potential mismatch in SLAs

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Should I use an internal API?

Pros

- Allow sharing of resources across systems/languages
- Can prevent isolated failures from bringing the entire system down
  - Only works if API consumers are fault tolerant
  - e.g. Netflix
- Can use to limit other systems' access to network elements (e.g. data sources)

Cons

- Additional overhead
- Additional maintenance
- More points of failure
- System coupling
- Less obvious discrepancy issues
  - Compile vs. runtime

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Decision factors for using an API

External API

- Can the same functionality be reasonably recreated and maintained?
- Is there a code library that performs a similar function?
- Is the API functionality vital to the success of the project?

Internal API

- Would multiple systems benefit from access to the resource?
  - Still could use a library if they're the same language
- Do the consuming systems have value without the resource?

At larger organizations, these questions are typically not answered (exclusively) by software developers.

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Either way, exercise caution

- Fault tolerance/graceful failure
- Caching/backup plan

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Examples from OU Credit Union

- Google Maps API
- Correio
- NeverBounce
  - We also have in-house solutions for integrating e-mail bounce detection with proprietary vendor software
- Visa
- Some vendors have SOAP APIs
- We expose some APIs internally to share implementations of defined processes and communicate between languages

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Examples of Cool (Free-ish!) APIs

- REST
  - Weather: darksky.net (aka forecast.io)
    - first 1,000 requests/day are free
  - Contact: twilio.com
    - Free trial with credit, some limitations on call length, etc.
  - Others can be found in this article: https://blog.rapidapi.com/most-popular-apis/
- SOAP
  - Dictionary: http://services.aonaware.com/DictService/DictService.asmx
  - E-mail Verification: http://ws.cdyne.com/emailverify/Emailvernotestemail.asmx

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY Credit Union

What makes good documentation?

- Clear communication layer requirements (request/response objects).
- Detailed descriptions (types and purpose) of properties.
- Easy to find and search, separated from other documentation.
- Language-agnostic
- Examples:
  - https://darksky.net/dev/docs (simple)
  - https://developers.google.com/maps/documentation/ (complex well organized)

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

If you have any questions, please check the FAQ and, if you can't find what you're looking for there, send us an email.

API Request Types

Forecast Request

```
https://api.darksky.net/forecast/[key]/[latitude],[longitude]
```

A Forecast Request returns the current weather conditions, a minute-by-minute forecast for the next hour (where available), an hour-by-hour forecast for the next 48 hours, and a day-by-day forecast for the next week.

Example Request

```
GET https://api.darksky.net/forecast/0123456789abcdef9876543210fedcba/42.3601,-71.0589

{
    "latitude": 42.3601,
    "longitude": -71.0589,
    "timezone": "America/New_York",
    "currently": {
        "time": 1509993277,
        "summary": "Drizzle",
        "icon": "rain",
        "nearestStormDistance": 0,
        "precipIntensity": 0.0089,
        "precipIntensityError": 0.0046,
        "precipProbability": 0.9,
        "precipType": "rain",
        "temperature": 66.1,
        "apparentTemperature": 66.31,
        "dewPoint": 60.77,
        "humidity": 0.83,
```

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Request Parameters

Required parameters slot directly into the request URL. Optional parameters should be specified as HTTP query parameters.

key required
Your Dark Sky secret key. (Your secret key **must** be kept secret; in particular, **do not** embed it in JavaScript source code that you transmit to clients.)

latitude required
The latitude of a location (in decimal degrees). Positive is north, negative is south.

longitude required
The longitude of a location (in decimal degrees). Positive is east, negative is west.

exclude=[blocks] optional
Exclude some number of data blocks from the API response. This is useful for reducing latency and saving cache space. The value `blocks` should be a comma-delimeted list (without spaces) of any of the following:

- `currently`
- `minutely`
- `hourly`
- `daily`
- `alerts`
- `flags`

extend=hourly optional
When present, return hour-by-hour data for the next 168 hours, instead of the next 48. When using this option, we **strongly** recommend enabling HTTP compression.

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Response Format

API responses consist of a UTF-8-encoded, JSON-formatted object with the following properties:

latitude required
The requested latitude.

longitude required
The requested longitude.

timezone (e.g. `America/New_York`) required
The IANA timezone name for the requested location. This is used for text summaries and for determining when `hourly` and `daily` data block objects begin.

offset deprecated
The current timezone offset in hours. (Use of this property will almost certainly result in Daylight Saving Time bugs. Please use `timezone`, instead.)

currently optional
A data point containing the current weather conditions at the requested location.

minutely optional
A data block containing the weather conditions minute-by-minute for the next hour.

hourly optional
A data block containing the weather conditions hour-by-hour for the next two days.

daily optional
A data block containing the weather conditions day-by-day for the next week.

alerts optional
An alerts array, which, if present, contains any severe weather alerts pertinent to the requested location.

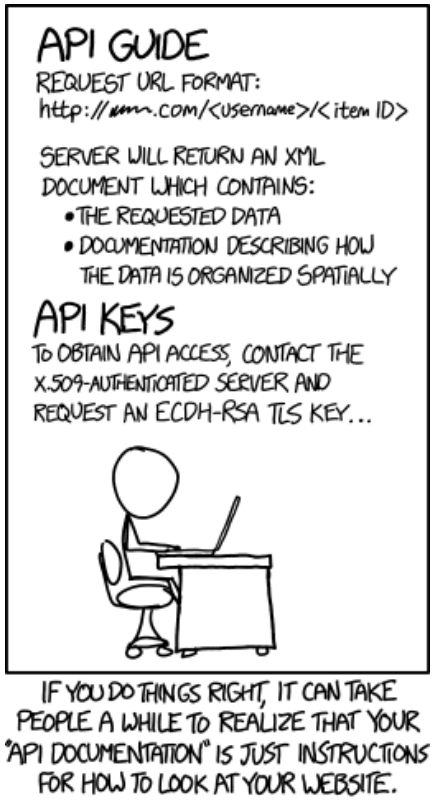
flags optional
A flags object containing miscellaneous metadata about the request.

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

This documentation is bad. What now?

- Commiserate.
  - When documentation for a public API is poor, odds are high that someone else has struggled with it and has advice to share. Check Stack Overflow and search for blog posts if you're having a hard time using an API.
- Eliminate variables.
  - Work in test environments and do trial and error testing.



https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

What We Covered

- API Definition
- Different types of API
- Different communication standards and protocols for Web APIs
- How to consume a REST API
- Considerations around using an API
- How to deal with poorly documented APIs

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

Questions?

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union

References

- Article by Glenn Johnson - https://it.toolbox.com/blogs/glennjohnson/application-programming-interface-api-definition-and-classification-by-types-040214
- Comics from xkcd.com
- The developer hive mind
- And, of course… Wikipedia

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY Credit Union

Thank you!

https://github.com/oucreditunion/grizzhacks2019

OAKLAND UNIVERSITY
Credit Union