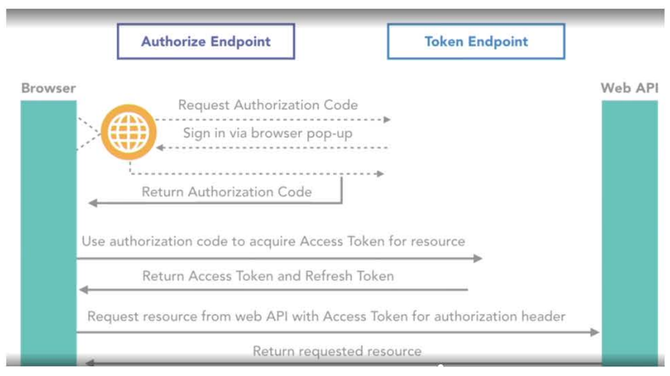
Authorization Code for Web Applicatin

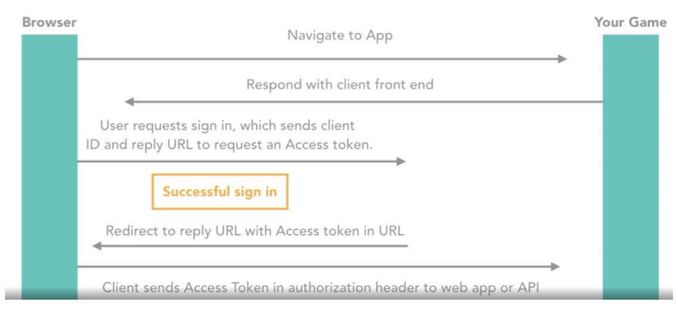
Let's move server side.

The Basics

- It's not simple, but it is reliable
- It's not mobile friendly, but we can trust it
- Works the same in any back-end programing language
- The attack/threat vectors are small and short-lived
- Large and growing ecosystem including libraries, Postman, etc.

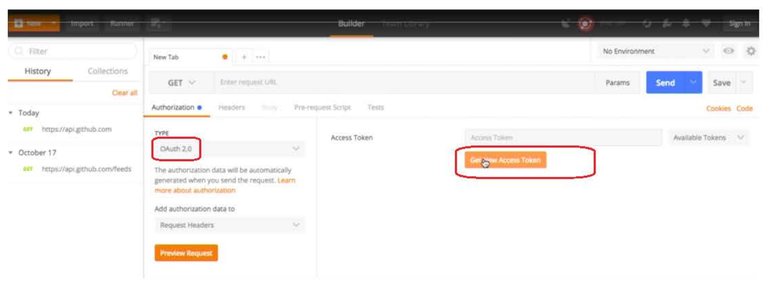


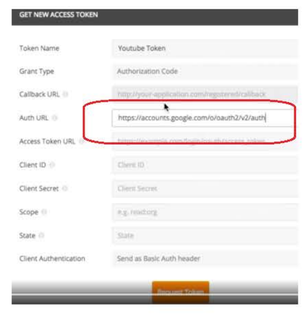
Facebook Authorize Endpoint

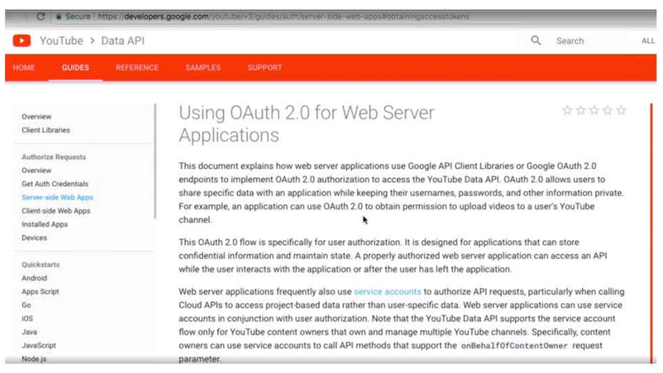


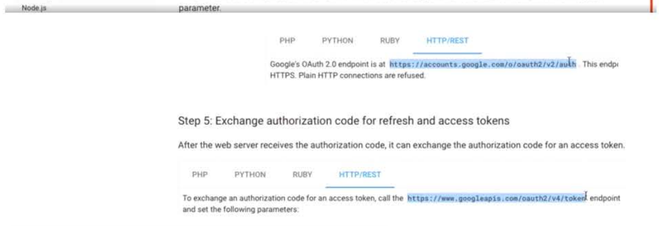
Creating the Authorization Code URL is complex.
Use a well-established library. Your life will be easier.

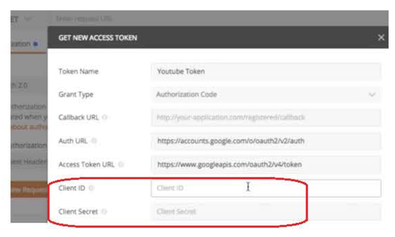
Build an example with JavaScript

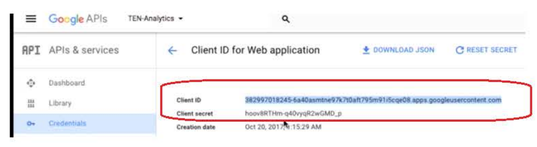


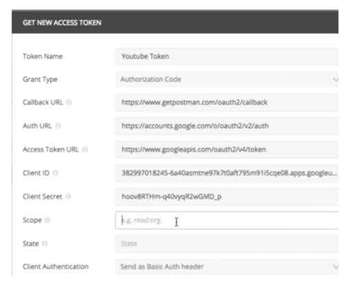


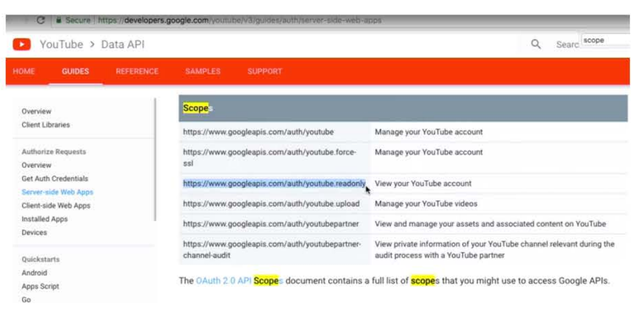


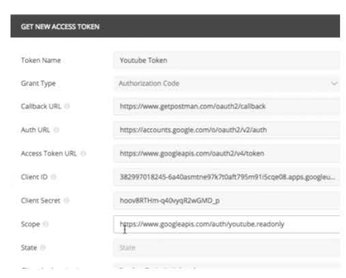


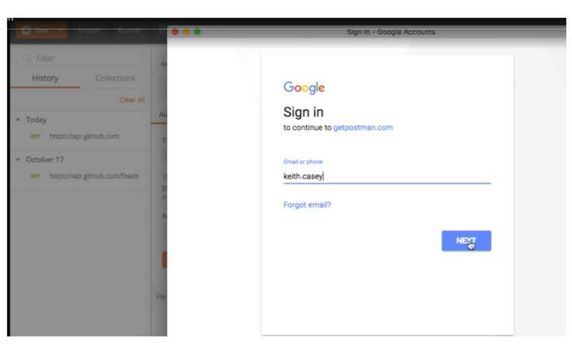


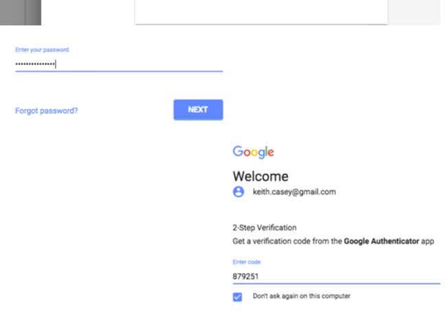


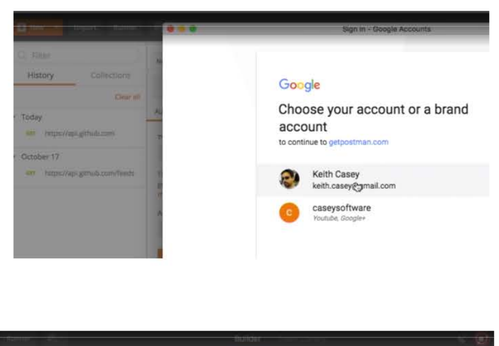


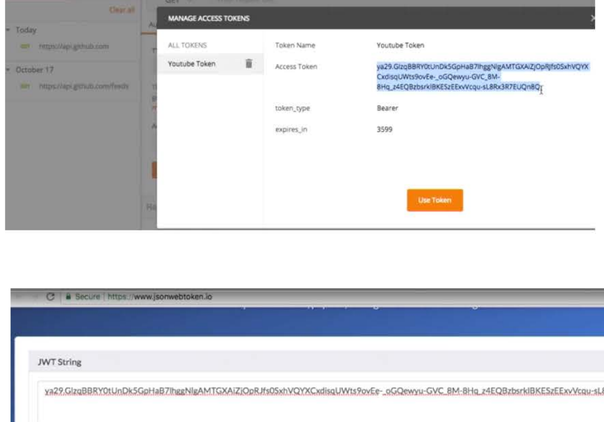












Authorization Code Flow Security Considerations

This is usually the most secure, except for...

Security Considerations

- Protect the auth code

  It's the one-time code that gives us an access token.

  (We never see the resulting access token.)

- Be careful of active browser sessions

  Generally, user-consent screens will mitigate this risk.

- Must use secure communications

Security Considerations

- Test your redirect_uri whitelist

  This is where the server sends the resulting access token.

  Confirm that your server doesn't accept any arbitrary redirect_uri.

- All resource servers must validate the access tokens

  No, don't argue—just do it.

Security Considerations

- Impersonation

  The May 2017 "Google Docs Worm"

  Disguised itself as Google and asked for permissions via the normal OAuth authorization flow

  It hit millions of people—no one is sure of the impact or goal.

  Google responded quickly and shut it down.

  This was 100% human behavior, not a flaw in OAuth.