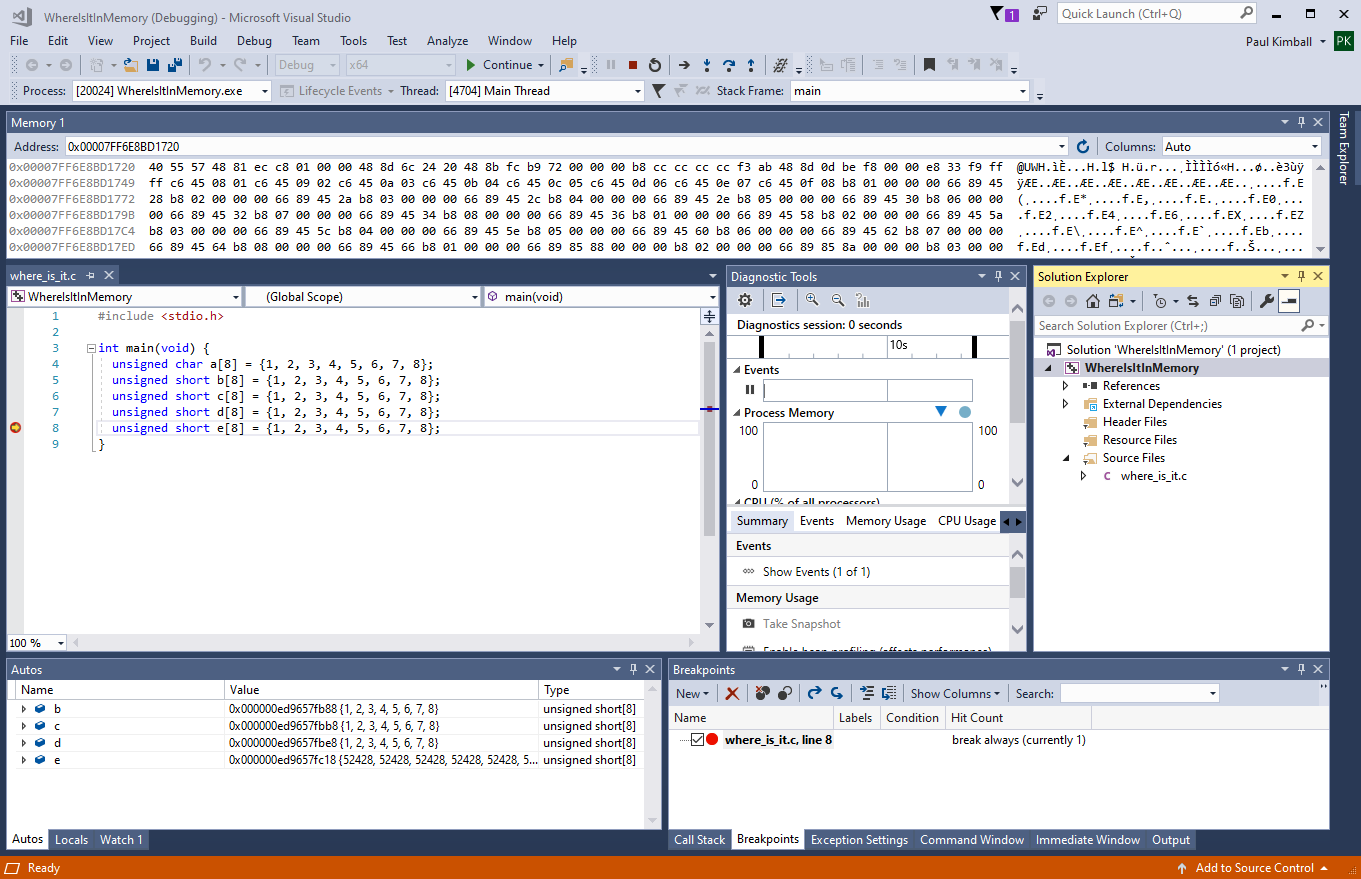
FCPA 2022  
 Software Setup

Thursday, January 6, 2022

* + - 1. Base Software Setup
* Windows 10
* Visual Studio 2017 (or later)
* Git for Windows with git-bash
* Ford GitHub account
* On-Line References
* Additional tools, libraries and extensions may be added during class
  + - 1. Short Walkthrough
* Visual Studio Customization
  + Workloads
  + Fonts, colors, editors
* Git for Windows
  + git-bash and cmd shells
* GitHub Personal Access Token

# VS 2017 User Interface



## Old-school, one tool does everything

### Many menus with many choices

### Tons of toolbars with tiny icons

### Many tabbed panes to display different tools, views, files

### Many optional plugins

### Hundreds of settings to control everything from fonts to compilers to code management systems

## Looks impressive... maybe even scary.

* + But mostly, cluttered
* I suggest you clean it up
  + Remove unused workloads and extensions
  + Customize UI for clarity

# Remove Unused Workloads

* A "workload" is a way of using VS to develop a certain kind of software product
  + Usually requires special tools, editors and utilites
* Visual Studio supports many programming languages, execution environments, and plugins

### *You don't need them all, especially not at the same time.*

## Tools menu -> Get Tools and Features ...

### Run the Visual Studio installer to add or remove official Microsoft workloads and options

#### Languages, Build systems, Runtime Environments (WSL, Azure, etc.)

* For this class we only need two workloads:
  + **Desktop development with C++**
  + **Linux development with C++**

### We will be writing "console apps" in C

#### You can execute them at the command line

#### No fancy windows or dialog boxes

# Extensions and NuGet

## Extensions menu -> Manage Extensions

### On VS2017 this is under Tools menu -> Extensions and Updates

### Check the VS Marketplace for nifty tools

#### Enhance UI features and integrate external tools

#### Color themes and plugins are here

## Tools menu -> NuGet Package Manager

### Add or remove .NET class libraries downloaded from NuGet

#### These apply to a project/solution

#### Cached on your system for reuse in other projects

* Don't worry about these for now

# Set Autosave Options

## Tools menu -> Options... [Alt-T O]

### And... BEHOLD ALL TEH OPTIONS !!!

#### Grouped in categories/subcategories

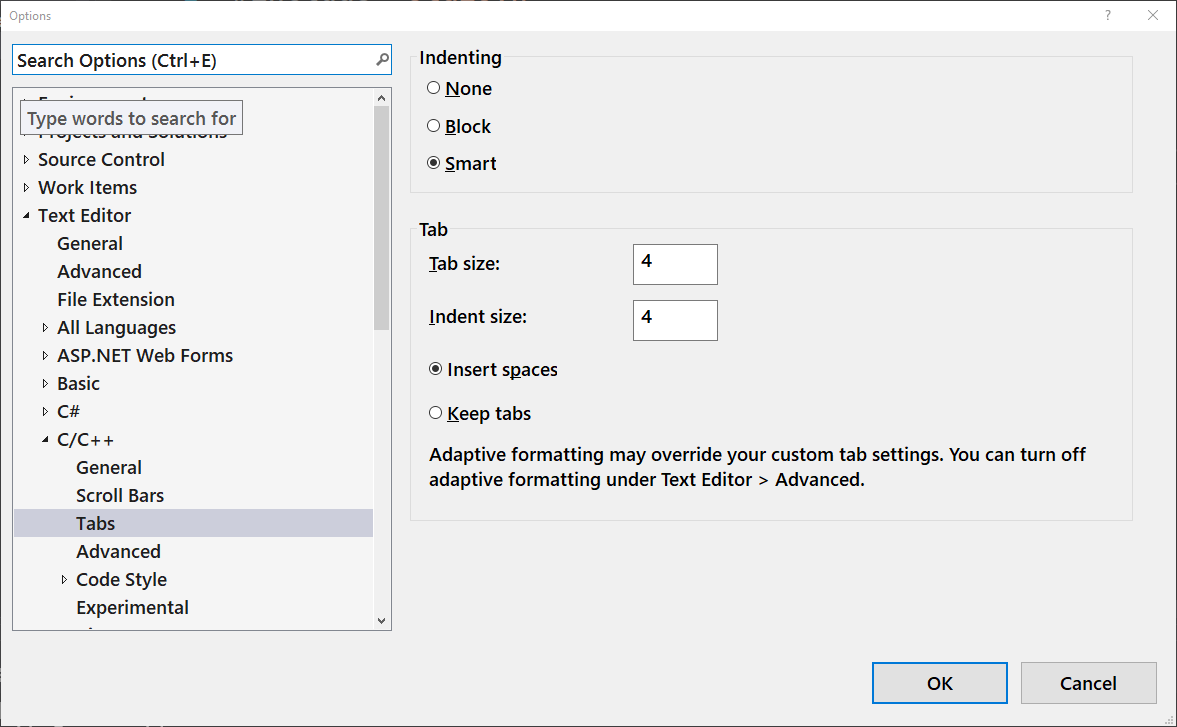
#### At least it has a "search" feature

### The "infinite options" dialog is the bane of all big IDEs

#### Eclipse and NetBeans are like this, too

* Make a cheatsheet for options you care about; google all else
  + Some categories, at least, will become familiar to you

## The options below may save you some time:



### **Environment / Documents**

#### Detect when file is changed [ON] - detects if you edited a file outside of VS

#### Reload modified files [ON] - reloads externally edited files

### **Environment / AutoRecover**

#### Save Autorecover information [ON] - enables periodic autosaves

### **Projects and Solutions / Locations**

#### Where to store your projects

# Personalize the UI

## Move the views to suit yourself

### Views and Toolbars are draggable, dockable, pinnable

## Select which views you want to see

### Show a view:

#### View menu -> select view

### Close a view

#### Select "X" icon in title or tab

#### Never be afraid to close a view; you can always get it back

## Set your UI theme

### Tools menu -> Options... [Alt-T O]

#### Category: **Environment / General**

#### Select **Color Theme** (it also saves your font changes)

#### Additional themes can be downloaded through the extension manager

## Choose fonts for editors, terminals, menus and other views

### Tools menu -> Options... [Alt-T O]

#### Category: **Environment / Fonts and Colors**

### Then, select "Show settings for:"

#### **Environment** - set font for UI menus, tabs, lists - (including Solution Explorer) May be called "Environment Font" on earlier releases

#### **Terminal** - set font for command prompts and shells

#### **Text Editor** - sets font for code editors

#### etc.

## Customize how the editor formats your code

### Tools menu -> Options... [Alt-T O]

#### Category: **Text Editor / C/C++**

### Then select options in subcategories

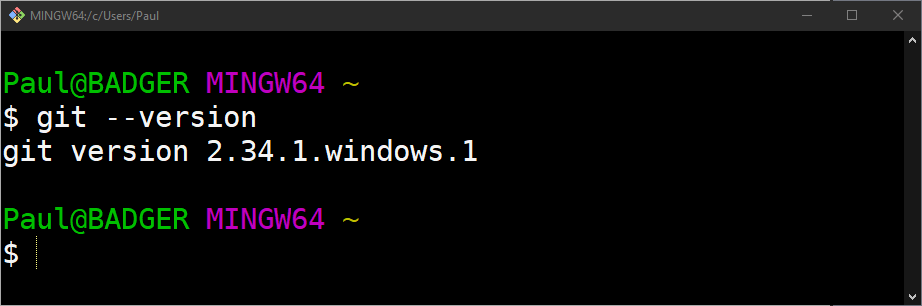
#### General / Line numbers [ON]

#### General / Word wrap [ON]

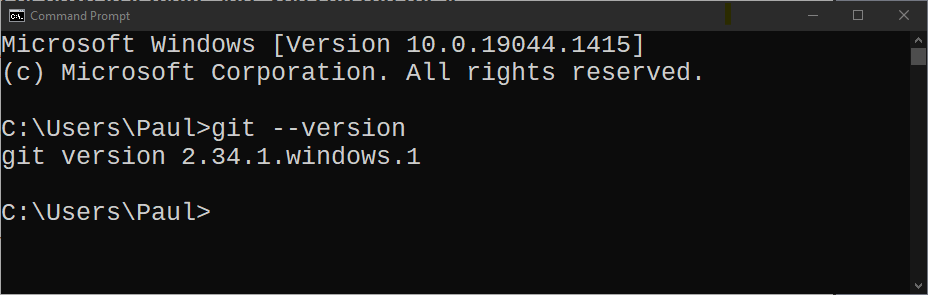
#### Code Style / Linter -> ... select your preferences

#### Code Style / Formatting -> ... select your preferences

* + - 1. bash and cmd shells
* There are two shells we'll use in class. We'll talk about them later. For now, just make sure you can run these from the start menu
* git-bash
  + UNIX bash shell ported to Windows



* cmd.exe
  + Windows command shell
  + We're not using PowerShell in class



* You can use git from either one

# Your GitHub account

## Lets you access the repositories that I will create during class

## Allows you to create and manage remote git repositories of your own

## When you use GitHub, you are subject to:

### Authentication

#### Prove who you are

### Authorization

#### Knowing who you are, here's what you can do

## To access GitHub via a web browser:

* + Go to https://github.ford.com

### Use your Ford User ID / Password to authenticate

## To access GitHub from the Git command line:

### You will need a Personal Access Token (PAT)

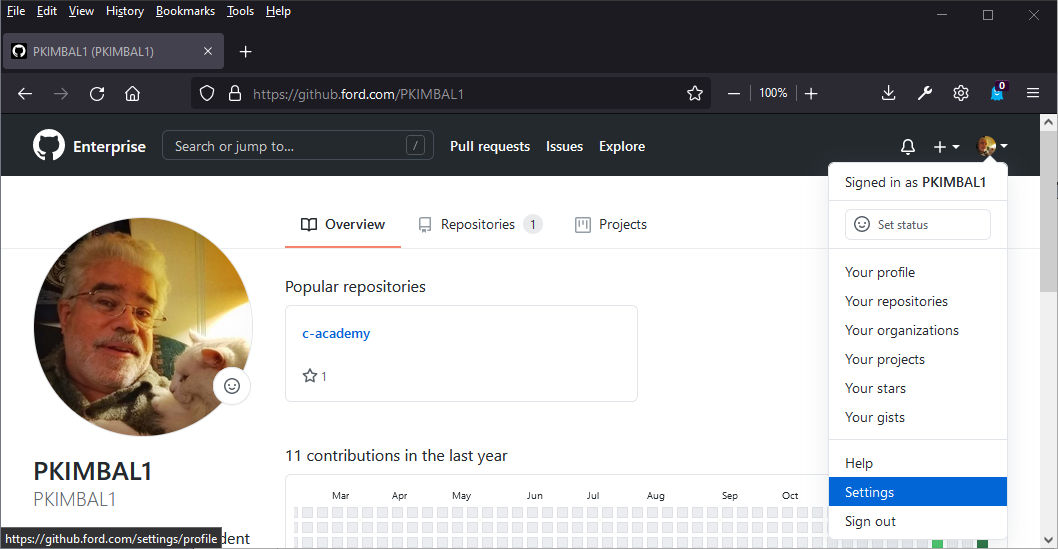
#### The PAT authorizes the git command to work as your agent

### You must log in to GitHub to generate a PAT

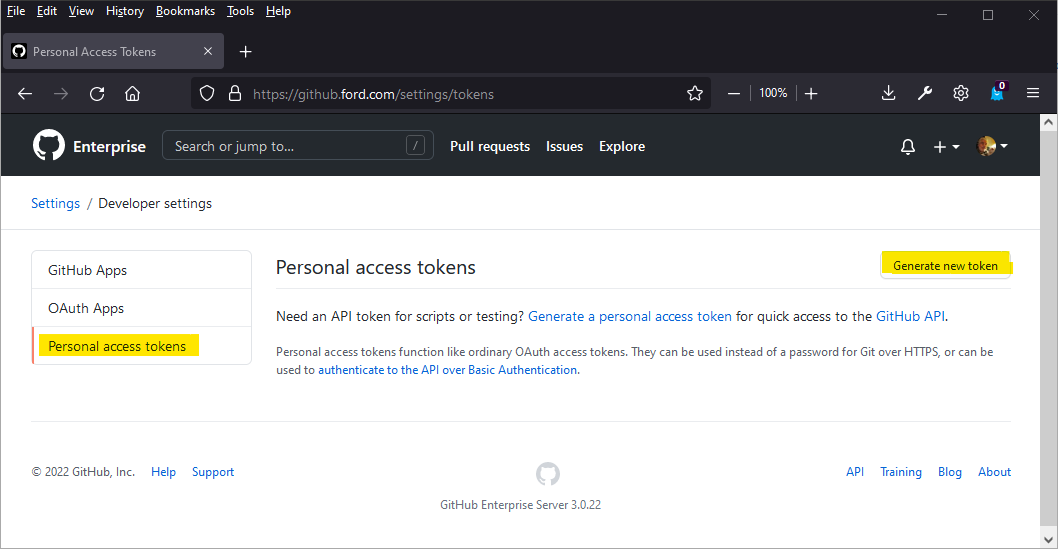
# Generate a PAT

## Log into GitHub and go to settings:

### Then Go to Developer settings (last one on the left)



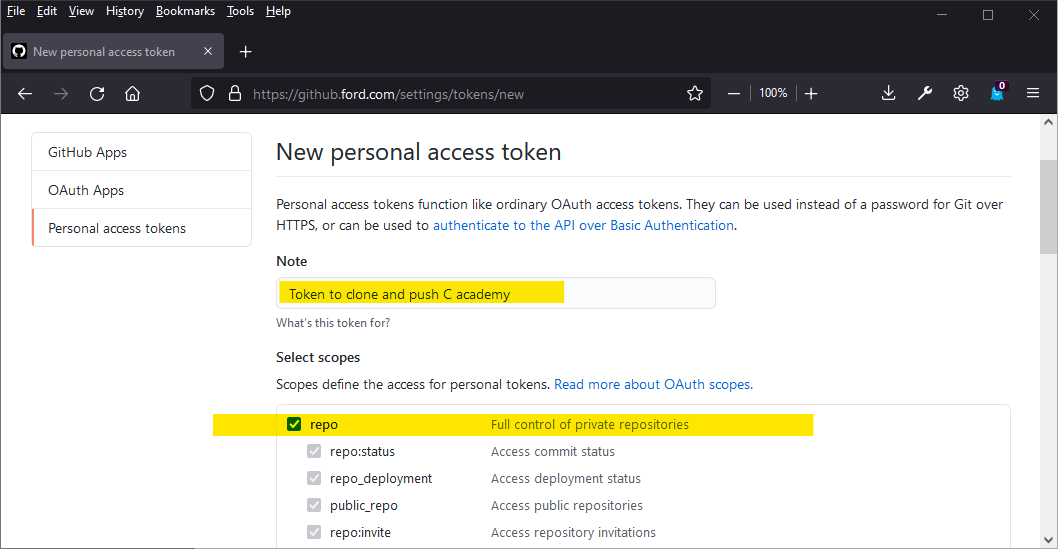
## Generate a token



# Generate a PAT

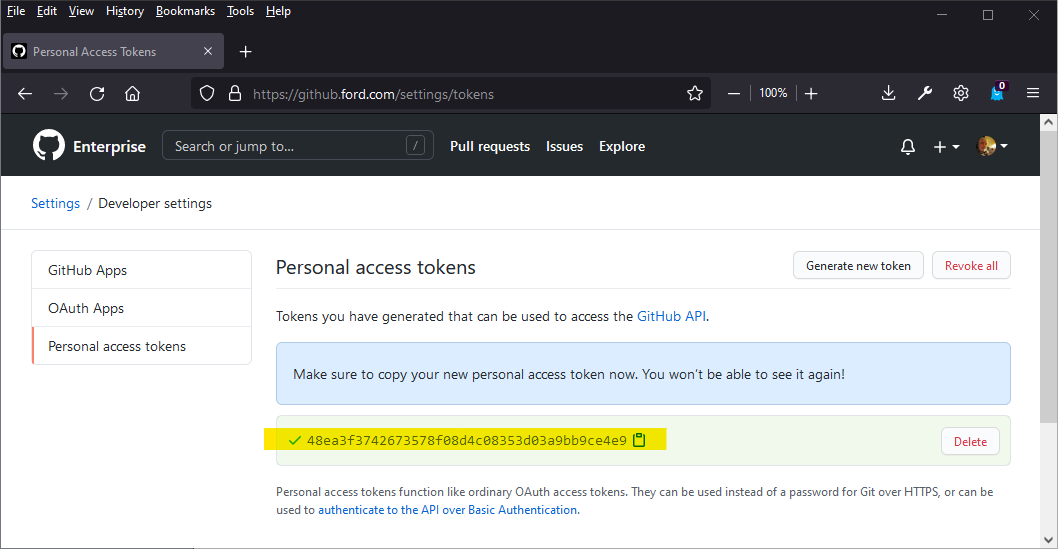
## Enter a note and select "repo" scope for authorization

### Then scroll down and click button to create token



## Make sure you copy it while you can and put it somewhere safe

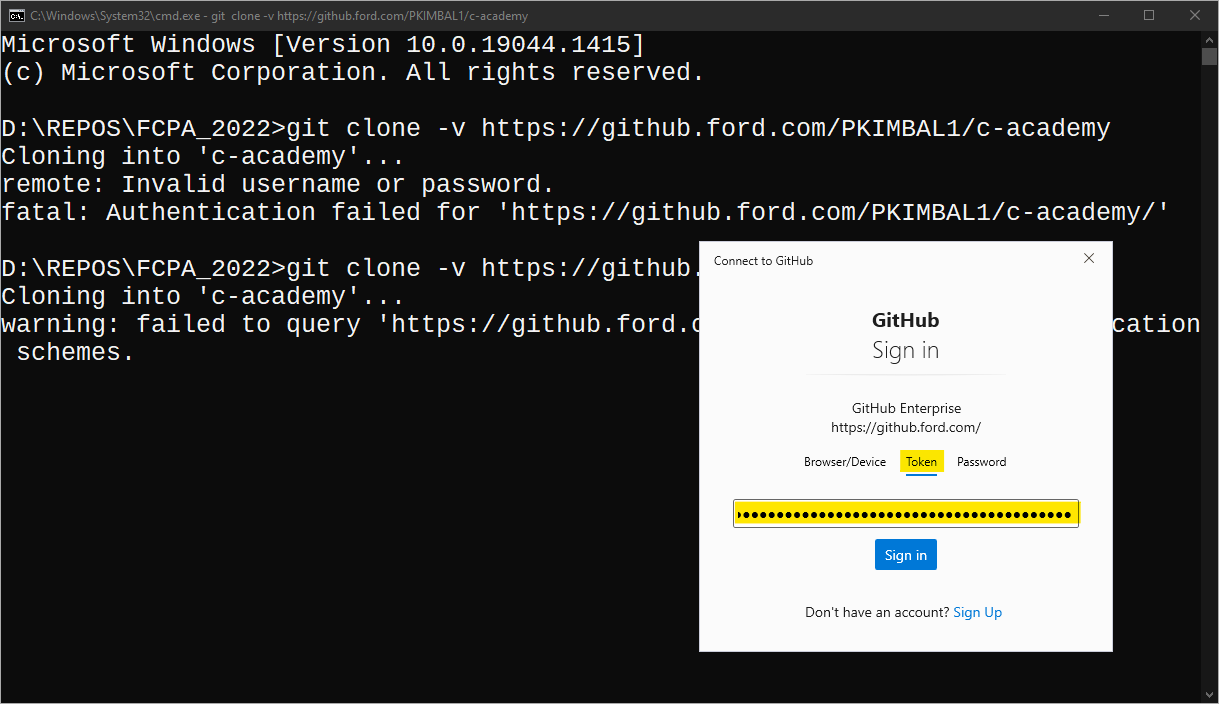
### You can click the little clipboard icon to copy



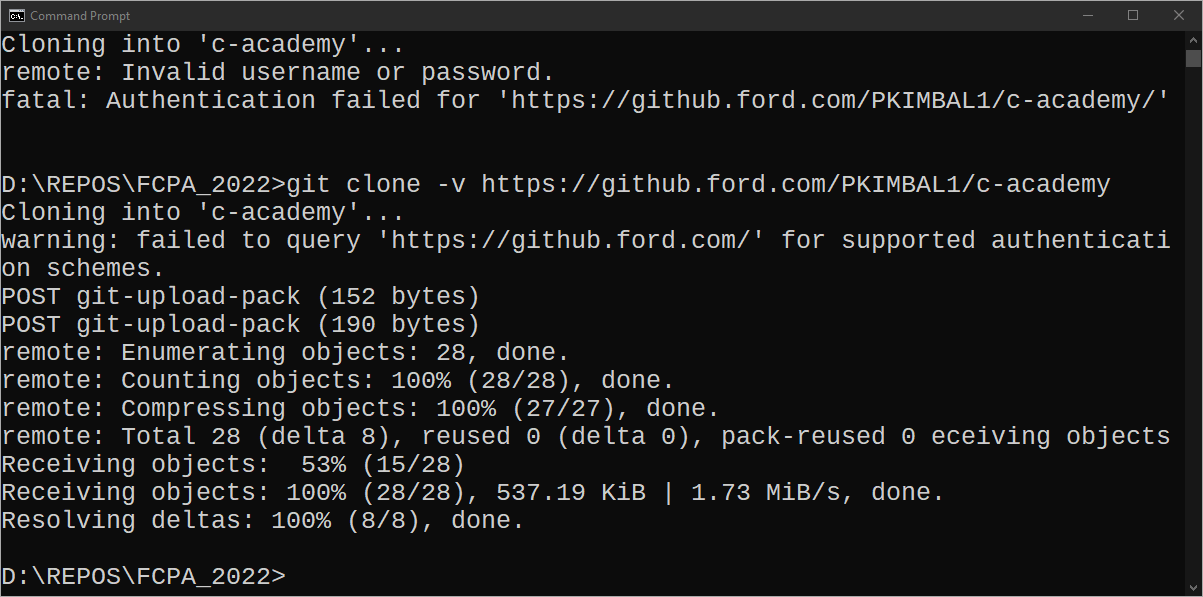
# Using your PAT

## Later on, when you use the git command to clone a repo:

### You might have to try this a couple times



## The token is good until it expires or is revoked



* + - 1. References
* Make sure that you can access the following online references
* cppreference.com
  + <https://cppreference.com>
* C17 Language Specification (pre-ballot draft)
  + <https://files.lhmouse.com/standards/ISO%20C%20N2176.pdf>
* Visual Studio Reference
  + <https://docs.microsoft.com/en-us/visualstudio/windows/>
* Microsoft Visual C++ Compiler
  + <https://docs.microsoft.com/en-us/cpp/c-language>
* MISRA 2012
  + <https://www.rlis.ford.com/cgi-bin/standards/restricted.cgi?p=misra&f=Ford-MISRA-C-2012>
* Books on RLIS
  + - 1. Any Questions?