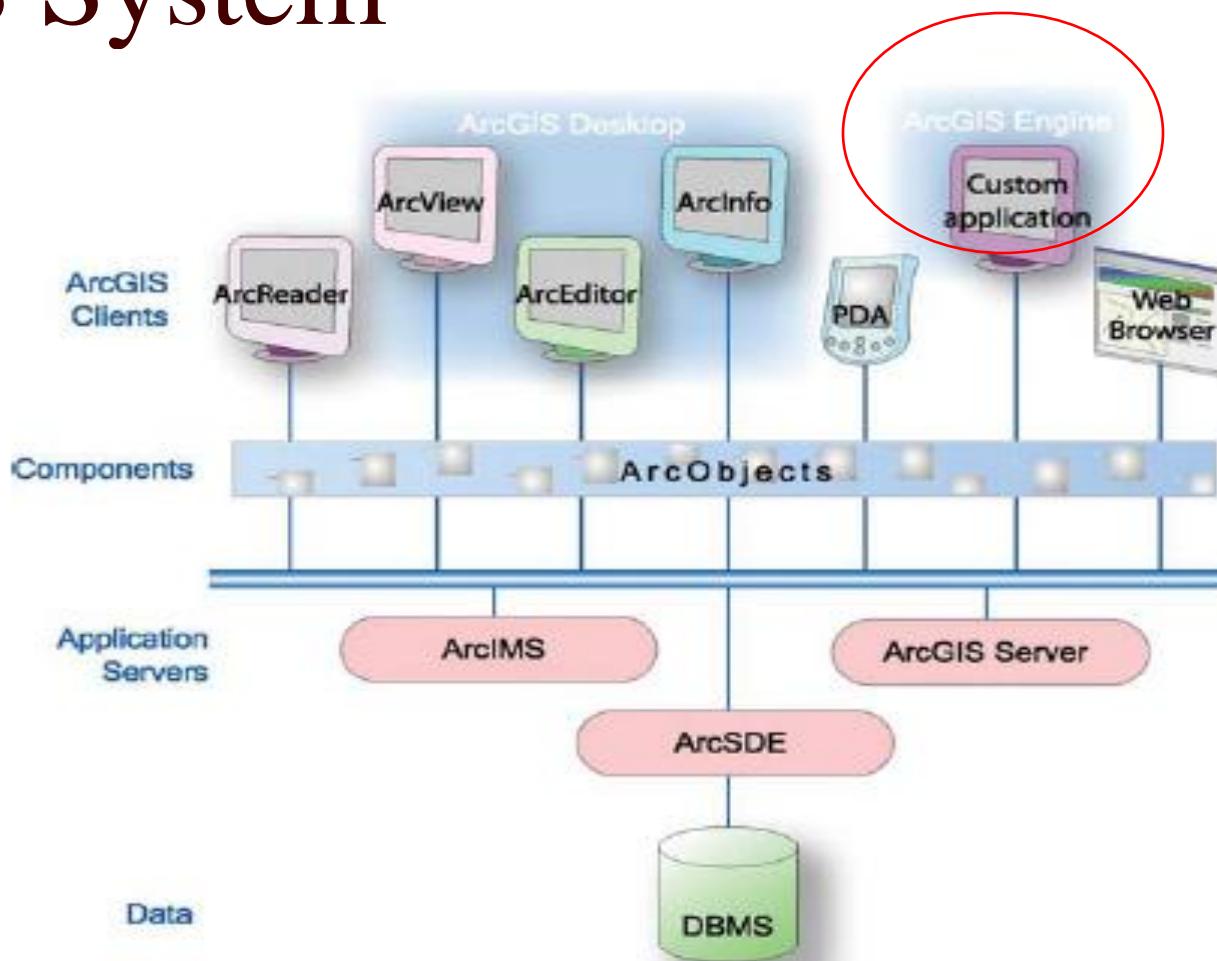


Writing Geoprocessing Scripts With ArcGIS

Lecture 10

GIS System



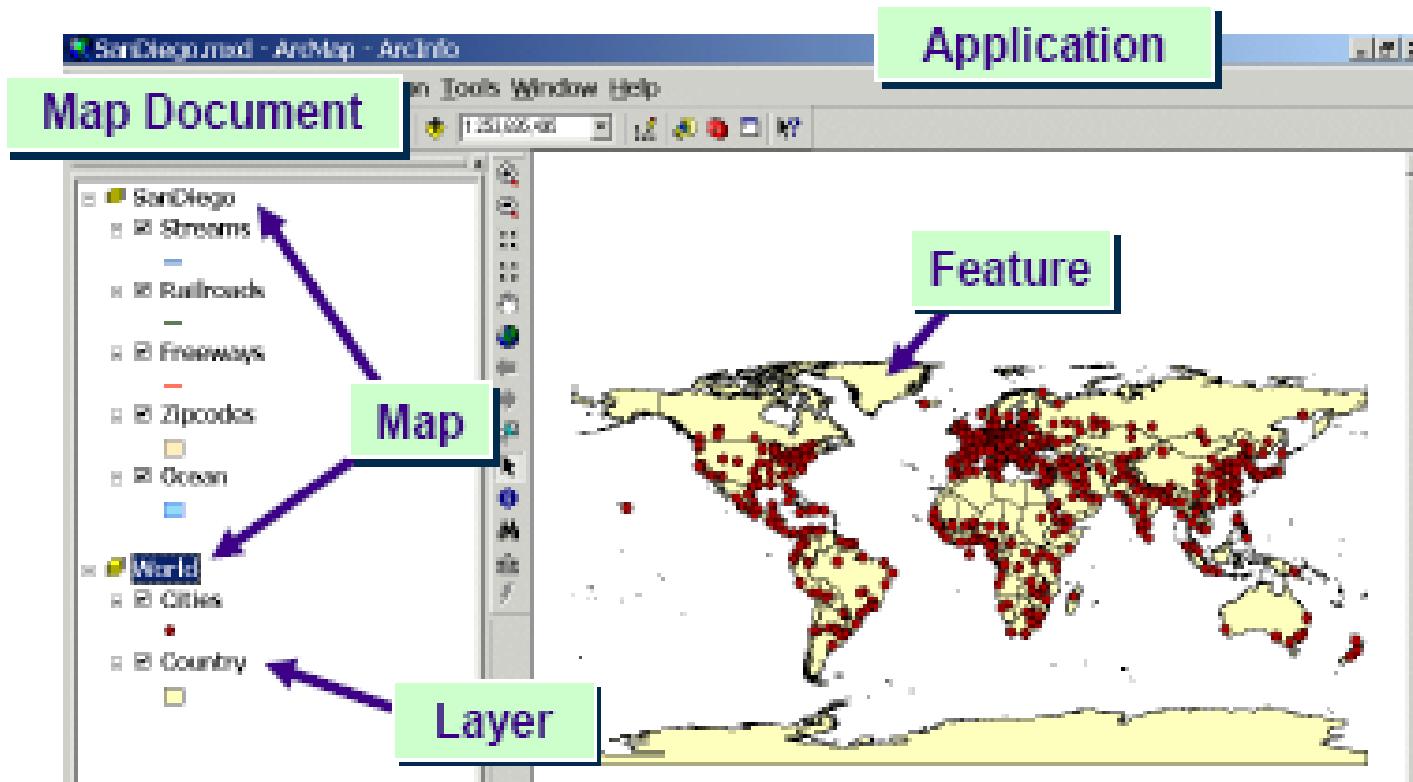
ArcSDE® Gateway is an interface for managing geodatabases in numerous relational database management systems (RDBMSs).

ArcObjects

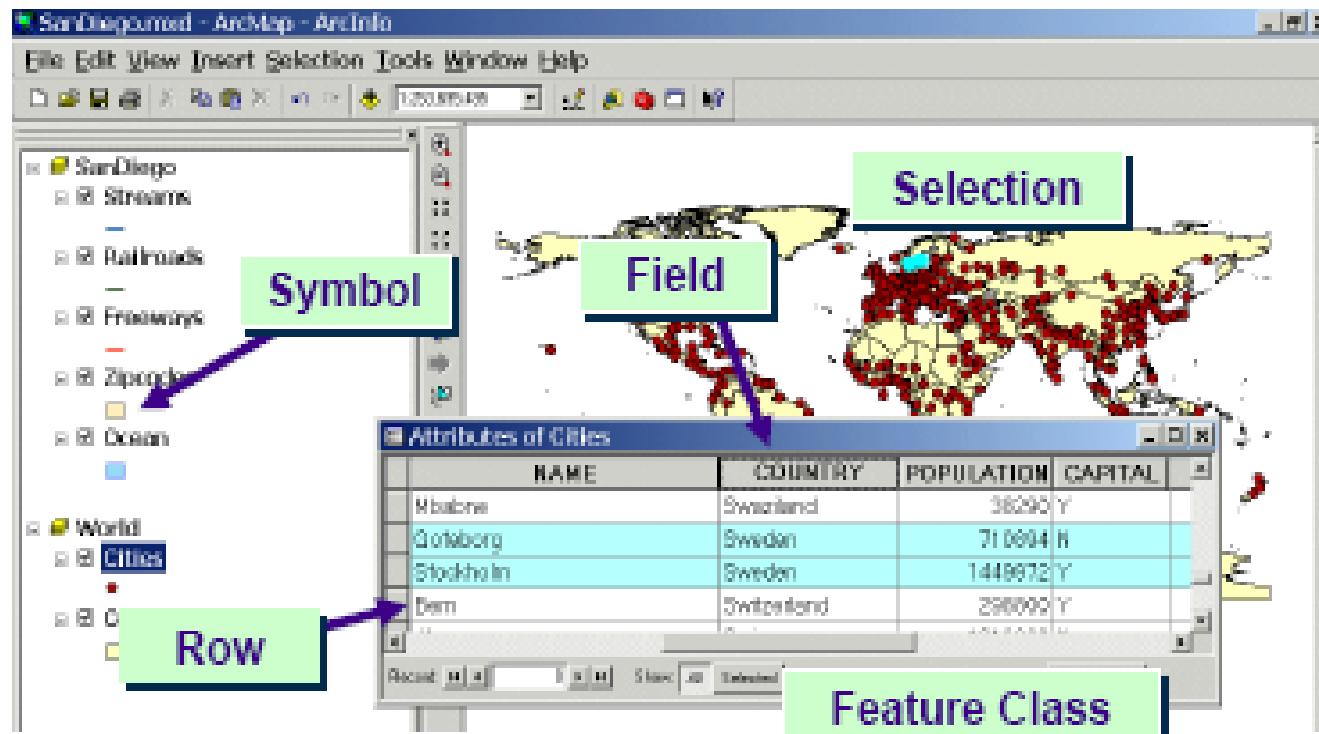
- ArcObjects are the building blocks of ArcGIS. With ArcObjects, you can create your own menus, tools, workflows, applications, and custom feature classes for use with ArcGIS.
- ESRI ArcObjects is the development platform for the ArcGIS family of applications, such as ArcMap, ArcCatalog, ArcScene, ArcGIS Engine, and ArcGIS Server. The ArcObjects software components expose the full range of functionality available in ArcInfo, ArcEditor, and ArcView to software developers
- Can use ~~VBA~~, Python, C++, Java to program

Common ArcObjects

- ◆ ArcGIS is built with a set of ArcObjects



Common ArcObjects



Interacting with ArcObjects

- ◆ Each ArcObject has properties and methods
 - ◆ Property: Characteristic of an object (a noun)
 - ◆ Method: Something the object knows how to do (a verb)
- ◆ Interact with objects through properties and methods

Map

Properties

- Layer count
- Name
- Spatial reference
- Map scale
- Extent

Methods

- Add layer
- Clear selection
- Select feature

Feature Class

Properties

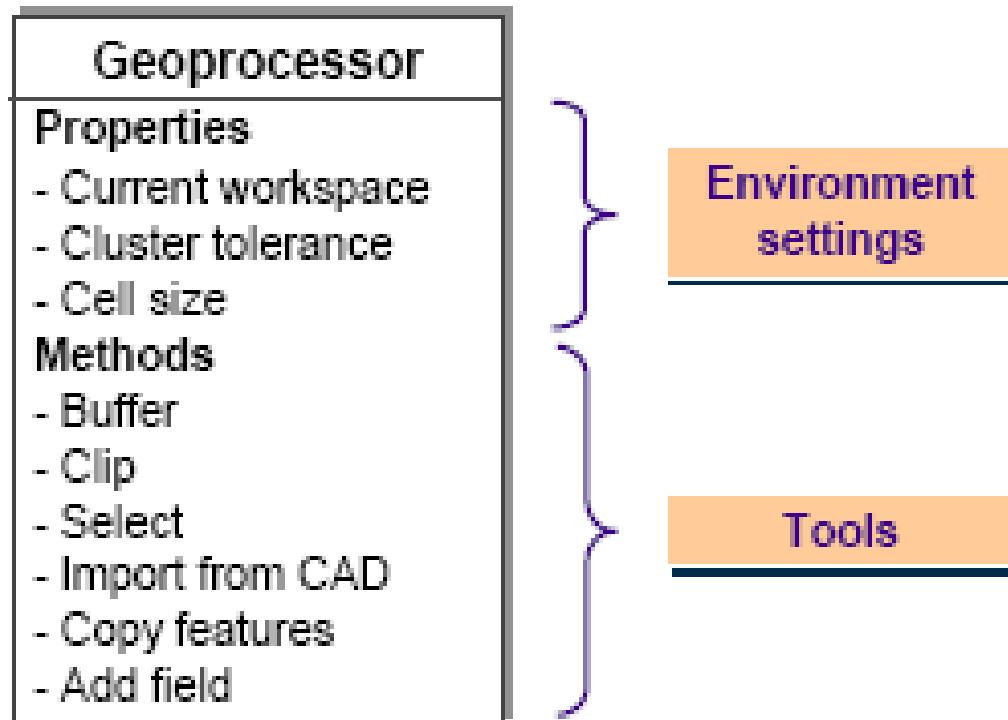
- Shape type
- Spatial reference
- Extent

Methods

- Create feature
- Remove feature

The Geoprocessor (GpDispatch) ArcObject

- ◆ Most geoprocessing functionality on one ArcObject
 - ◆ Geoprocessor or GpDispatch
- ◆ The Geoprocessor has many properties and methods



Accessing the Geoprocessor from Python

- ◆ Geoprocessor can be used in many languages
 - ◆ Perl, VBScript, JScript, Python, VBA, VB, C#, and so on
 - ◆ Any COM compliant language

For ArcGIS 10 and Python 2.6.5:

```
import arcpy
```

Tells Python to import basic ArcGIS geoprocessing functionality

```
from arcpy import env
```

Tells Python to import ability to control the ArcGIS Environment

```
from arcpy.sa import *
arcpy.CheckOutExtension("Spatial")
```

Example: Tells Python to import functionality from ArcGIS Spatial Analyst
and a second command to get/check the Spatial Analyst license

Syntax for properties and methods

- ◆ To assign a value to a property

Object.Property = Value

```
env.workspace = "C:/Temp"
```

- ◆ To get the value of a property

Object.Property

```
print "The name of the workspace is " + env.workspace
```

- ◆ To use a method

Object.Method (arg, arg, ...)

```
arcpy.Buffer_analysis ("C:/input/roads.tif", "C:/output/Output.gdb/buffer_output", 100)
```

- ◆ Parentheses around arguments

- ◆ Arguments separated by commas

Toolbox aliases

- ◆ Always suffix the tool with the toolbox alias



`arcpy.Clip_arc()`



`arcpy.Clip_analysis()`



Always assign names
to custom toolboxes

The Select tool

◆ Syntax

```
Select_analysis (in_features, out_feature_class,  
    where_clause)
```

◆ Example

```
env.workspace = "c:/basedata/roads.gdb"
```

```
arcpy.Select_analysis("nfroads", "paved", '[ROAD_CLASS] = "PAVED"')
```

◆ Notes

Python uses forward slashes, different than Windows using back slashes

Double quotes pass text strings

Single quotes contain text strings intended to pass variable names

The Where clause Syntax

- ◆ SQL expression

- ◆ Dependant on the underlying database

- ◆ Shape file

"NAME" = 'Toronto'

- ◆ Geodatabase

[NAME] = 'Toronto'

- ◆ The easiest way

- * Open the Select tool, or the Select By Attributes dialog, and
 - * Copy the syntax that these dialogs use.

Python Basics

Variables in Python

- ◆ **Variables are dynamically typed**

- ◆ No declaration keyword
 - ◆ No type assignment

```
fc = "P:/Qiu/PYTH/Database/World/Cities.shp"
```

- ◆ **Variables are case sensitive**

```
scale = 10000  
Scale = 20000
```

Two different variables

- ◆ **Variables can hold different data types**

- ◆ Strings, numbers, lists, tuples, dictionaries, files

Strings

- ◆ Variables can hold strings

```
folder = "C:/Student"  
whereClause = "[STREET_NAM] = 'CATALINA'"
```

- ◆ Strings surrounded in double ("") or single ('') quotes

- ◆ Can embed one string in another

- ◆ Pathnames use ~~two back (\)~~ or one forward (/) slash

- ◆ \ is a reserved escape character and a line continuation character.

- ◆ Strings can be combined together

```
gdbPath = "C:/SouthAfrica.mdb"  
fc = "Roads"  
fullPath = gdbPath + "/" + fc
```

- ◆ Strings are indexed, 0-based from the left and 1-based from right

```
fc = "Street.shp"  
newFC = fc[:-4]  
  
fc[0] ---> "S" # S is in the 0 position  
  
fc[1:3] ---> "tr" # the last position is never included  
  
fc[:-4] ---> "Street" # get rid of the last 4 characters
```

Numbers and lists

- ◆ Variables can hold numbers and expressions

```
num1 = 1.2  
num2 = 3 + 5
```

- ◆ Variables can hold lists

```
numList = [1, 2, 3]  
fcList = ["Roads", "Streets", "Parcels", "Zipcodes"]
```

- ◆ Lists are indexed, 0-based from the left and 1-based from right

```
fc1 = fcList[1]      --> "Streets"
```

Lists are zero-based, position one is "Streets".

```
fc2 = fcList[0:2]    --> "Roads", "Streets"
```

The last position is not included.

```
fc3 = fcList[0:-1]   --> "Roads", "Streets", "Parcels"
```

Take values from zero to minus one. Last position not included.

```
fc4 = fcList[2:]     --> "Parcels", "Zipcodes"
```

Take all values from position two to the end.

Variable index

Word =
+---+---+---+---+---+
| H | e | l | l | o |
+---+---+---+---+---+
0 1 2 3 4 5
-5 -4 -3 -2 -1

- Word[0]='H'
- Word[2:4]='lp'
- Word[:3]='Hel'
- Word[-2:-4]='el'
- Word[-3:]= 'lpA'

Variable naming conventions

- ◆ Upper case versus lower case

- ◆ First word lower case, capitalize each successive word

```
outputFieldName = "City"
```

- ◆ Acronym at the beginning, use lower case letters

```
gdbPath = "C:/Stockholm.mdb"
```

```
fc = "Railroads.shp"
```

- ◆ Acronym in the middle or at the end, use upper case letters

```
inputFC = "Streets.shp"
```

- ◆ Avoid special characters (e.g. / \ & * !)

- ◆ Use descriptive variable names

Line continuation

- ◆ Line continuation characters

- ◆ Parentheses (), brackets [], and braces { }
- ◆ Backslash \

- ◆ Indentation is automatic

```
# Date: May 3, 2004
# Purpose: To buffer the eighth feature class in the list.

fcList = ["Freeways", "Climate", "Ocean", "MajorAttractions",
          "Railroads", "Mines", "Zipcodes", "Streams"]

distanceValues = 100, 200, 300, 400, 500, 600, 700, 800,
                 900, 1000, 1100, 1200, 1300, 1400, 1500

arcpy.Buffer_analysis (fcList[7], "BuffStreams1000",
                      distanceValues[9])
```

Built-in functions

- ◆ Python has many built-in functions

- ◆ `len()` – Returns the length

```
fc = "Railroads.shp"  
len(fc) --> 13  
fields = ["OID", "Shape", "Name"]  
len(fields) --> 3
```

- ◆ `max()` – Returns the maximum value

```
xExtent = (6260474.996464, 6338807.996464)  
max(xExtent)
```

- ◆ `open()` – Opens a file

```
coord = open("C:/XY.txt", "r").read()
```

- ◆ `round()` – Rounds a number

```
yCoord = 1811884.623964  
round(yCoord)
```

Accessing Python modules

- ◆ Most functions need to be imported from modules

- ◆ The math module

```
import math  
  
*math.sqrt(64) # Prefix the name of the function with the name of the  
module  
*math.pi
```

- ◆ The string module

```
import string  
string.split("-155.3 -43.5")  
string.upper("C:/student")
```

- ◆ The os.path module

```
import os.path  
os.path.basename("C:/student/Streets.shp") returns "Streets.shp"
```

```
os.path.dirname("C:/student/Streets.shp") returns "C:/Student"
```

Statements

- ◆ Python has many **statements**. Statements do not return values.

- ◆ **print** – Sends output to the Interactive Window

```
print "Hello"
```

- ◆ **import** – Imports a module

```
import math  
import string
```

- ◆ Other statements (discussed throughout the class)

```
if..elif..else  
while  
for..in  
try..except
```

Decision making syntax

- ◆ Testing conditions

```
if x == 1:  
    print "x is 1"  
elif x == 2:  
    print "x is 2"  
else:  
    print "x is not 1 or 2"
```

- ◆ Colons used at end of each condition

- ◆ Indentation defines what executes for each condition

- ◆ Python automatically indents when you press Enter
 - ◆ Use tabs or spaces, must be consistent

- ◆ Two equal signs for conditions, one for assignment

```
x = 1 # assignment  
y = 8 + 2 # assignment  
if x == 6: # testing a condition, not assigning a value
```

Looping syntax

- ◆ While loops, counted loops, and list loops

```
x = 5
while x < 10:
    print x
    x = x + 1

for x in range(1,5): #The last value is not executed.
    print x

x = [1, 2, 3]
for a in x:
    print a
```

- ◆ Colons used at end of each statement

- ◆ Indentation defines what executes for the loop

Case sensitive rules

◆ Case sensitive

◆ Functions and statements

max len open print import if Correct

Max LEN OpEn Print import IF Incorrect

◆ Variable names

Scale ≠ scale Not equal

◆ Not case sensitive

◆ Path names

"C:/STUDENT" = "c:/StUdEnT" Equal

◆ Geoprocessing properties and methods

arcpy.BUFFER = arcpy.buffer Equal

Introduction to Python and ArcGIS for Geoprocessing Scripts

Lecture 10

Python: An open-source programming language

Python Programming Language – Official Website - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Python Programming Language... +

www.python.org DuckDuckGo (SSL)

python

ABOUT >>

NEWS >>

DOCUMENTATION >>

DOWNLOAD >>

下載 >>

COMMUNITY >>

FOUNDATION >>

CORE DEVELOPMENT >>

Help

Package Index

Quick Links (2.7.2)

» Documentation

» Windows Installer

» Source Distribution

Quick Links (3.2.2)

» Documentation

» Windows Installer

» Source Distribution

Python Jobs

Python Merit Badge

Python Wiki

Python Insider Blog

Python 2 or 3?

Help Maintain Website

Help Fund Python

PayPal DONATE or VISA MASTERCARD

Advanced Search

search

Python Programming Language – Official Website

Python is a programming language that lets you work more quickly and integrate your systems more effectively. You can learn to use Python and see almost immediate gains in productivity and lower maintenance costs.

Python runs on Windows, Linux/Unix, Mac OS X, and has been ported to the Java and .NET virtual machines.

Python is free to use, even for commercial products, because of its OSI-approved open source license.

New to Python or choosing between Python 2 and Python 3? Read [Python 2](#) or [Python 3](#).

The Python Software Foundation holds the intellectual property rights behind Python, underwrites the [PyCon](#) conference, and funds other projects in the Python community.

[Read more, -or- download Python now](#)

» Second release candidates for Python 2.6.8, 2.7.3, 3.1.5, and 3.2.3 released

Another iteration of release candidates for Python 2.6.8, 2.7.3, 3.1.5, and 3.2.3 have been released for testing. They include several security fixes.

Published: Sun, 18 March 2012, 22:00 -0500

» CfP: PyCon Taiwan 2012

PyCon Taiwan 2012 deadline for talk proposals is April 2. Conference will be 6/9-10 in Taipei.

Published: Fri, 16 March 2012, 09:00 -0700

» IronPython 2.7.2 released

IronPython 2.7.2 has been released.

Published: Mon, 12 March 2012, 08:00 -0700

» CfP: EuroSciPy 2012

EuroSciPy 2012 deadline for talk proposals is April 30. Conference will be 8/23-27 in Brussels.

Published: Sat, 10 March 2012, 17:00 -0800

» First alpha for Python 3.3.0 released

The first alpha release for Python 3.3.0 have been released for testing.

Python 3 Poll

I wish there was Python 3 support in

(enter PyPI package name)

[Vote](#) [Results](#)

Carmanah lights the way with Python

Welcome

Introducing the EverGLN 1500-Parking Lot Light

All the magic of a standard ground

... joining users such as Rackspace, Industrial Light and Magic, AstraZeneca, Honeywell, and many others.

What they are saying...

Firaxis Games:

"Python, like many good technologies, soon spreads virally throughout your development team and finds its way into all sorts of applications and tools. In other words, Python begins to feel like a big hammer"

Many places for help, not just Python.org

FrontPage - PythonInfo Wiki - Mozilla Firefox

File Edit View History Bookmarks Tools Help

FrontPage - PythonInfo Wiki +

wiki.python.org/moin/ DuckDuckGo (SSL)

 python™

» FrontPage

» BeginnersGuide/Overview » BeginnersGuide » NonProgrammers » FrontPage

The Python Wiki

 Python is a great object-oriented, interpreted, and interactive programming language. It is often compared (favorably of course 😊) to Lisp, Tcl, Perl, Ruby, C#, Visual Basic, Visual Fox Pro, Scheme or Java... and it's much more fun.

Python combines remarkable power with very clear syntax. It has modules, classes, exceptions, very high level dynamic data types, and dynamic typing. There are interfaces to many system calls and libraries, as well as to various windowing systems. New built-in modules are easily written in C or C++ (or other languages, depending on the chosen implementation). Python is also usable as an extension language for applications written in other languages that need easy-to-use scripting or automation interfaces.

Getting Started

Beginners Guide	Documentation
Links to tutorials, courses and resources	Learning materials, topic guides and links to central resources
Beginner Errors	Python Books
Some common pitfalls of beginners	Books about Python plus reviews
Asking for Help	Python Audio Materials
Questions asked by beginners, answered here	A mixture of introductory and topical material
Languages	Python Implementations
Resources written in languages other than English	Different software which runs programs in the Python language

See also the documentation category for all known documentation-related pages.

Events, Courses, Conferences, Community

- » [Python Conferences](#) - information about the Python conference scene
- » [Python Events](#) - covers conferences, training courses and more
- » [Local User Groups](#) - find a Python group near you
- » [Participating in the Community](#) - where people using and producing Python get together

Support comes in many languages

The screenshot shows a Mozilla Firefox window with the title "Languages - PythonInfo Wiki - Mozilla Firefox". The address bar displays "wiki.python.org/moin/Languages". A red circle highlights this address bar. The main content area shows the "Languages" page of the PythonInfo Wiki. The left sidebar has sections for FRONTPAGE, RECENTCHANGES, FINDPAGE, HELPCONTENTS, and LANGUAGES, with LANGUAGES being the active tab. The right sidebar includes a search bar and buttons for "titles" and "text". The main content discusses language support and provides a numbered list of guidelines for maintaining language pages. Below this is a section titled "Languages" listing various language names in English and their native scripts.

» Languages

Attempt to have languages and links listed in the native tongue of the user.

Ideally, all the pages should be like the Polish or Turkish pages - all native language, only the necessary English.

There are some groundrules, some laid down by the site admins, some my suggestions:

- 1) Pages must be named in ASCII and English ([PolishLanguage](#))
- 2) Pages must have an explanation in English at the top (Links to Python information in <language X>)
- 3) (my suggestion) We probably want to limit invites to edit the pages to people we know well, or Pythonistas with a track record. Hopefully this is inclusive enough without opening the site up to a spam flood and vandalismfest.

Where these pages really need help:

- 1) check links, remove broken ones.
- 2) add new links that are quality Python information and active.
- 3) some care for languages that have next to nothing, but do have people in the Python community - even a link to the Wikipedia page for Python, in that language, is a start (Some are pretty complete and of high quality - the Russian language Wikipedia page for Python, for instance, packs a lot in).

Languages

- » AfrikaansLanguage Afrikaans
- » AlbanianLanguage Shqip
- » AmharicLanguage አማርኛ
- » ArabicLanguage العربية
- » ArmenianLanguage Հայերեն
- » AssameseLanguage অসমীয়া
- » AzerbaijaniLanguage Azərbaycan dilı
- » BelorussianLanguage Беларуская мова
- » BengaliLanguage বাংলা

Search, and you will find many guides

BeginnersGuide - PythonInfo Wiki - Mozilla Firefox

File Edit View History Bookmarks Tools Help

BeginnersGuide - PythonInfo Wiki +

wiki.python.org/moin/BeginnersGuide DuckDuckGo (SSL)

 python™

» **BeginnersGuide**

» BeginnersGuide/Overview » NonProgrammers » FrontPage » BeginnersGuide

Beginner's Guide to Python

New to programming? Python is free and easy to learn if you know where to start! This guide will help you to get started quickly.

[Chinese Translation](#)

New to Python?

Read [BeginnersGuide/Overview](#) for a short explanation of what Python is.

Getting Python

Next, install the Python interpreter on your computer. This is the program that reads Python programs and carries out their instructions; you need it before you can do any Python programming. Mac OSX distributions from 10.3 (Panther) and up, include a version of Python, which, although it can be as much as two years out of date, may be suitable for beginning. Linux distributions also frequently include Python and it is readily upgraded.

There are currently two major versions of Python available: Python 2 and Python 3. The [Python2orPython3](#) page provides advice on how to decide which one will best suit your needs. At the time of writing (21 Jun 2010), the rest of this page assumes you've decided to use Python 2.

See [BeginnersGuide/Download](#) for instructions for downloading the correct version of Python.

At some stage, you'll want to edit and save your program code. Take a look at [HowToEditPythonCode](#) for some advice and recommendations.

Learning Python

Next, read a tutorial and try some simple experiments with your new Python interpreter.

» If you've never programmed before, see [BeginnersGuide/NonProgrammers](#) for a list of suitable tutorials.

» If you have previous programming experience, consult [BeginnersGuide/Programmers](#), which lists more advanced tutorials.

» If English isn't your first language, you might be more comfortable with a tutorial that's been translated into your language. Consult [python.org's list of Non-English resources](#).

Most tutorials assume you know how to run a program on your computer. If you are using Windows and need help with this, see [How do I Run a Program Under Windows](#).

Python as a first language

BeginnersGuide/NonProgrammers - PythonInfo Wiki - Mozilla Firefox

File Edit View History Bookmarks Tools Help

BeginnersGuide/NonProgrammers +

wiki.python.org/moin/BeginnersGuide/NonProgrammers DuckDuckGo (SSL)

python TM

» BeginnersGuide NonProgrammers

» BeginnersGuide/Overview » FrontPage » BeginnersGuide » NonProgrammers

Python for Non-Programmers

If you've never programmed before, the tutorials on this page are recommended for you; they don't assume that you have previous experience.

If you have previous programming experience, the list of programmer-oriented tutorials on the [BeginnersGuide/Programmers](#) page may get you started more quickly, but the tutorials on this page may still be helpful.

» [How to Think Like a Computer Scientist - 2nd edition](#) Allen Downey's open source textbook has a Python version, written with Jeff Elkner. It's also available in book form. It was updated and current version is 2nd Edition.

» [The Programming Historian](#) From the "About This Book" page: "This book is a tutorial-style introduction to programming for practicing historians. We assume that you're starting out with no prior programming experience and only a basic understanding of computers. More experience, of course, won't hurt. Once you know how to program, you will find it relatively easy to learn new programming languages and techniques, and to apply what you know in unfamiliar situations."

» [Learning to Program](#) An introduction to programming for those who have never programmed before, by Alan Gauld. It introduces several programming languages but has a strong emphasis on Python.

» [Learn Python The Hard Way](#) The title is a misnomer. It would be better titled "Learn Python By Coding It." The author determined that learning python should be similar to learning an instrument. You don't get a book on scales, but you're taught a scale and practice it. The author teaches you how to code properly, how to think like a programmer, and develop quality problem solving skill through a set of 52 exercises that build on each other. .

» [A Byte of Python](#), by Swaroop C.H., is also an introductory text for people with no previous programming experience.

» [Invent Your Own Computer Games with Python, 2nd Ed](#), by Al Sweigart is a free e-Book that teaches complete beginners how to program by making games.

» [Making Games with Python & Pygame](#), by Al Sweigart is a free e-Book that introduces the Pygame framework for novices and intermediate programmers to make graphical games.

» [One Day of IDLE Toying](#) A very gentle introduction to the IDLE development environment that comes with Python. This tutorial by Danny Yoo has been translated into nine different languages.

» [Instant Hacking](#) A minimal crash course by Magnus Lie Hetland that's an excellent starting point.

» Free Python video lectures are also available as a course titled [Intro to programming with Python and Tkinter](#), Unix users can view the video using mplayer once you have downloaded the files. Windows users will need to have a DivX player, available from <http://www.divx.com/divx/windows/>. (One user reports success viewing the videos on OS X 10.4 using the VLC player -- <http://www.videolan.org/>)

» The [Young Programmers Podcast](#) contains video lessons on Python, Pygame, Jython, Scratch, Alice, Java and Scala.

» [A Non-Programmer's Tutorial for Python 2.6](#) by Josh Cogliati.

Python for Programmers

BeginnersGuide/Programmers - PythonInfo Wiki - Mozilla Firefox

File Edit View History Bookmarks Tools Help

BeginnersGuide/Programmers ... +

wiki.python.org/moin/BeginnersGuide/Programmers DuckDuckGo (SSL)

 python™

» BeginnersGuide Programmers

» BeginnersGuide/Overview » FrontPage » BeginnersGuide » NonProgrammers » Programmers

Python for Programmers

The tutorials on this page are aimed at people who have previous experience with other programming languages (C, Perl, Lisp, Visual Basic, etc.). Also of potential interest are such related Beginners Guides as [BeginnersGuide/Overview](#) and [BeginnersGuide/NonProgrammers](#), and the tips in [MovingToPythonFromOtherLanguages](#).

- » [Python Tutorial](#) This tutorial is part of Python's documentation set and is updated with each new release.
- » [Basic to Advanced Tutorial](#) A good tutorial on Python especially for the beginners.
- » [Python Essential Reference \(book\)](#) If you want a highly compressed K&R-style 'just the facts' overview, David Beazley's "Python Essential Reference" covers practically all of the language in about a hundred pages.
- » [Wikibook:Python Programming](#)
- » [Wikiversity:Python](#) The Wikianything information about Python.
- » [Instant Python](#) A minimal crash course by Magnus Lie Hetland.
- » [Google's Python Class](#) - Google's Python tutorial for "people with a little bit of programming experience"
- » [Python 2 & 3 Quick-Guides](#) A fast and efficient guide, with many examples, for quickly learning many tricks of Python.
- » [Python Programming for Beginners](#) A short introduction to writing command-line applications in Python by Jacek Artymiaik.
- » [Python 101 - Beginning Python](#) and [Python 201 - \(Slightly\) Advanced Python](#) Two self-training courses from Dave Kuhlman. Python 101 introduces the basic data types, and 201 covers particular tasks such as parsing text and writing unit tests.
- » [Python Short Course](#) A set of course slides by Richard P. Muller of Caltech that are aimed at scientific users. For example, the first example is a script to process output from a quantum chemistry simulation.
- » [Learn Python in 10 minutes](#)
- » [Python for Science](#) Course material written by Konrad Hinsen for an introduction to Python aimed at biologists.
- » [Software Carpentry](#): lectures aimed at scientists and engineers.
- » You can find a variety of Python tutorials on [Code Teacher](#).
- » [ComparingTypes](#) A quick look at some common programming types for python and other languages
- » [Python for Programmers](#) - for "Professional programmers who need to learn Python "
- » A basic [Python Tutorial](#) that goes into some interesting features and examples.
- » [Python Koans](#) Learn Python through TDD
- » [Intro to Python](#) - A Brief Presentation about Python mainly aimed at experienced programmers. Might be nice as a first pass over the language.

As with any language: “Read the freaking manual”

The screenshot shows a Mozilla Firefox browser window displaying the Python Documentation Index at www.python.org/doc/. The page features the Python logo and navigation links for "ABOUT", "NEWS", "DOCUMENTATION", "CURRENT DOCS", "LICENSE", "HELP", "FAQS", "BEGINNER'S GUIDE", "WIKI", "PEP INDEX", "NEW-STYLE CLASSES", "REGULAR EXPRESSIONS", and "AUDIO/VISUAL TALKS". A red circle highlights the "DOCUMENTATION" link in the sidebar. The main content area includes sections for "Python Documentation", "Note" (about a documentation project for Python packages), "Python 2.x" (with links to various documentation modules like What's new in Python 2.7, Tutorial, Library Reference, etc.), and "Python 3.x" (with links to Python 3.2.2 Documentation). At the bottom, there are donation links for PayPal and credit cards, and a "Non-English Resources" section.

Python Documentation Index - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Python Documentation Index +

www.python.org/doc/ DuckDuckGo (SSL)

python™

» Documentation Advanced Search

ABOUT >>

NEWS >>

DOCUMENTATION >>

Current Docs

License

Help

FAQs

Beginner's Guide

Wiki

PEP Index

New-Style Classes

Regular Expressions

Audio/Visual Talks

DOWNLOAD >>

下載 >>

COMMUNITY >>

FOUNDATION >>

CORE DEVELOPMENT >>

Python Wiki

Python Insider Blog

Python 2 or 3?

Help Maintain Website

Help Fund Python

or

Non-English Resources

» Documentation

Python Documentation

Python's standard documentation is substantial; download your own copy or browse it online! See also the complete list of documentation by Python version. If you're not sure where to go, try the [help page](#).

Note

A new documentation project, which will be merged into the Python documentation soon, covers creating, installing and distributing Python packages:

- [The Hitchhiker's Guide to Packaging \(creating and distributing Python packages\)](#)

Python 2.x

- Browse Current Documentation - (Module Index)
 - [What's new in Python 2.7](#)
 - [Tutorial](#)
 - [Library Reference](#)
 - [Language Reference](#)
 - [Extending and Embedding](#)
 - [Python/C API](#)
 - [Using Python](#)
 - [Python HOWTOs](#)
- [Search the online docs](#)
- [Download Current Documentation](#) (multiple formats are available, including typeset versions for printing.)

Python 3.x

- Browse Python 3.2.2 Documentation - (Module Index)
 - [What's new in Python 3.2](#)
 - [Tutorial](#)
 - [Library Reference](#)

ArcGIS 10 installs Python 2.6.5

Python Documentation by Version - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Python Documentation by Vers... +

www.python.org/doc/versions/ DuckDuckGo (SSL)

python™

» Documentation > Python Documentation by Version

ABOUT >>

NEWS >>

DOCUMENTATION >>

Current Docs

License

Help

FAQs

Beginner's Guide

Wiki

PEP Index

New-Style Classes

Regular Expressions

Audio/Visual Talks

DOWNLOAD >>

下載 >>

COMMUNITY >>

FOUNDATION >>

CORE DEVELOPMENT >>

Python Wiki

Python Insider Blog

Python 2 or 3?

Help Maintain Website

Help Fund Python

 or 



Non-English Resources

search Advanced Search

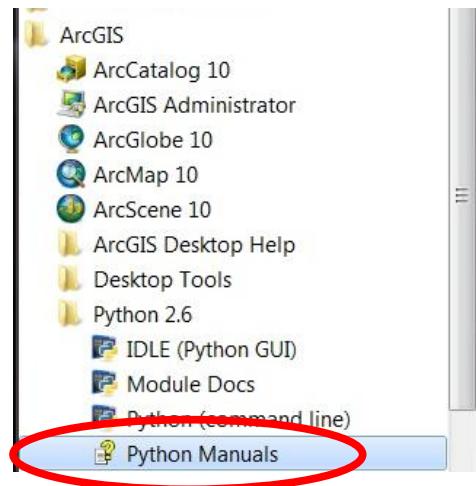
Python Documentation by Version

Some previous versions of the documentation remain available online. Use the list below to select a version to view.

- Unreleased, development versions of the documentation. This may contain more information than the current released documentation.
- Python 3.2.2, documentation released on 4 September 2011.
- Python 3.2.1, documentation released on 10 July 2011.
- Python 3.2, documentation released on 20 February 2011.
- Python 3.1.4, documentation released on 11 June 2011.
- Python 3.1.3, documentation released on 27 November 2010.
- Python 3.1.2, documentation released on 21 March 2010.
- Python 3.1.1, documentation released on 17 August 2009.
- Python 3.1, documentation released on 27 June 2009.
- Python 3.0.1, documentation released on 13 February 2009.
- Python 3.0, documentation released on 3 December 2008.
- Python 2.7.2, documentation released on 11 June 2011.
- Python 2.7.1, documentation released on 27 November 2010.
- Python 2.7, documentation released on 4 July 2010.
- Python 2.6.7, documentation released on 3 June 2011.
- Python 2.6.6, documentation released on 24 August 2010.
- **Python 2.6.5, documentation released on 19 March 2010.** (This item is circled in red.)
- Python 2.6.4, documentation released on 25 October 2009.
- Python 2.6.3, documentation released on 2 October 2009.
- Python 2.6.2, documentation released on 14 April 2009.
- Python 2.6.1, documentation released on 4 December 2008.
- Python 2.6, documentation released on 1 October 2008.
- Python 2.5.4, documentation released on 23 December 2008.
- Python 2.5.3, documentation released on 19 December 2008.
- Python 2.5.2, documentation released on 21 February 2008.
- Python 2.5.1, documentation released on 18 April 2007.
- Python 2.5, documentation released on 19 September 2006.

Only use ArcGIS 10 help / tutorials / examples

Version 9.3 is significantly different



Only use Python 2.6.5 help / tutorials / examples

Upgrading might break link with ArcGIS

The screenshot shows a Mozilla Firefox window displaying the Python v2.6.5 documentation. The title bar reads "Overview — Python v2.6.5 documentation - Mozilla Firefox". The address bar shows "docs.python.org/release/2.6.5/". The main content area is titled "Python v2.6.5 documentation". A sidebar on the left contains links for "Download", "Docs for other versions" (listing Python 2.7, 3.1, 3.2, and Old versions), and "Other resources" (FAQs, Guido's Essays, New-style Classes, PEP Index, Beginner's Guide, Book List, Audio/Visual Talks, Other Doc Collections). A "Quick search" bar is at the bottom left. The main content area includes sections for "Parts of the documentation:", "Indices and tables:", and various documentation links like "What's new in Python 2.6?", "Tutorial", "Using Python", "Library Reference", "Language Reference", "Python HOWTOs", "Extending and Embedding", "Python/C API", "Installing Python Modules", "Distributing Python Modules", "Documenting Python", "FAQs", "Search page", and "Complete Table of Contents".

Overview — Python v2.6.5 documentation - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Overview — Python v2.6.5 docu... +

docs.python.org/release/2.6.5/ DuckDuckGo (SSL)

Python v2.6.5 documentation » modules | index

Download

Download these documents

Docs for other versions

Python 2.7 (in development)
Python 3.1 (stable)
Python 3.2 (in development)
Old versions

Other resources

FAQs
Guido's Essays
New-style Classes
PEP Index
Beginner's Guide
Book List
Audio/Visual Talks
Other Doc Collections

Quick search

Enter search terms or a module, class or function name. Go

Python v2.6.5 documentation

Welcome! This is the documentation for Python 2.6.5, last updated Mar 19, 2010.

Parts of the documentation:

What's new in Python 2.6?
or all "What's new" documents since 2.0

Tutorial
start here

Using Python
how to use Python on different platforms

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python HOWTOs
in-depth documents on specific topics

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

Installing Python Modules
information for installers & sys-admins

Distributing Python Modules
sharing modules with others

Documenting Python
guide for documentation authors

FAQs
frequently asked questions (with answers!)

Indices and tables:

Global Module Index
quick access to all modules

General Index
all functions, classes, terms

Search page
search this documentation

Complete Table of Contents
list all sections and subsections

ESRI created Python routines to automate GIS

The screenshot shows the ArcGIS 10 Help Library interface. The title bar is redacted. The menu bar includes 'Hide', 'Locate', 'Back', 'Home', 'Options', and 'Resource Center'. The left sidebar contains 'Contents', 'Favorites', and 'Search' buttons, followed by a table of contents with sections like 'Welcome to ArcGIS Help', 'What's new in ArcGIS', 'Essentials Library', 'Professional Library', 'Administrator Library', 'GIS glossary', 'Copyright information', 'License agreement', and 'Acknowledgements'. The main content area displays the 'Welcome to the ArcGIS Help Library' page, which states: 'Welcome to the help library for the ArcGIS system. This help library has been compiled to provide comprehensive documentation for using all aspects of ArcGIS. The goal is to address the needs of a number of key audiences:'. It lists three audiences: 'GIS practitioners' (mapping, data management, editing, analysis, geoprocessing), 'Developers' (.NET, Java, C++, Web programming APIs, SQL), and 'Administrators' (installing, managing ArcGIS software, DBAs, Web server environments). Below this is a table comparing four libraries based on audience:

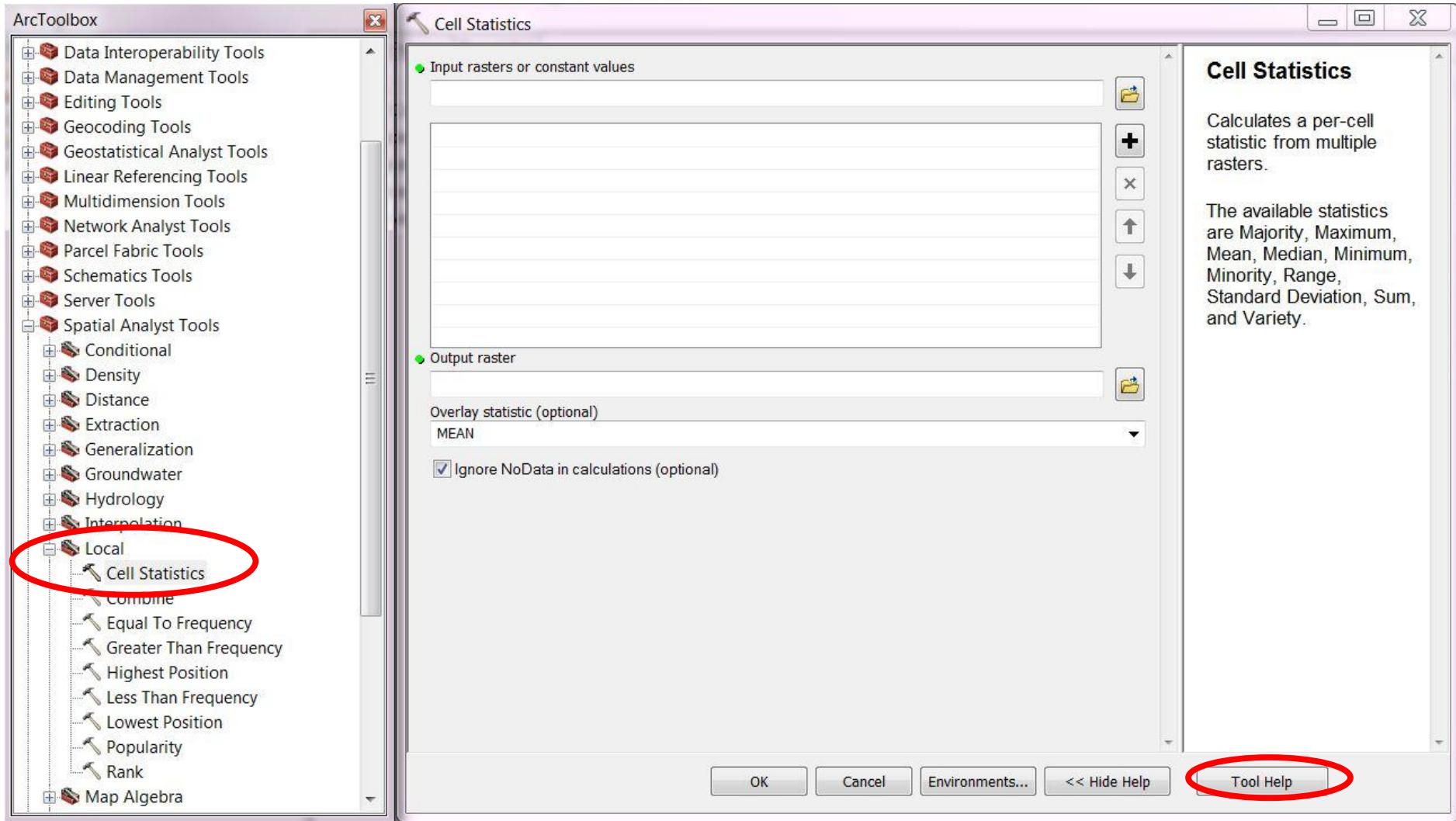
Library	Audience
Essentials Library provides an introduction to GIS and ArcGIS. These topics provide a foundation for using ArcGIS in your daily work. These topics are directed at all audiences and are useful for users who are new to GIS or ArcGIS. This library is also very helpful for seasoned ArcGIS users.	All users
Professional Library provides comprehensive help topics for GIS professionals and covers all aspects of using the ArcGIS system and many of the most common GIS workflows.	GIS professionals
ArcGIS includes a series of Developer SDKs , which can be accessed from the ArcGIS Resource Center. Each SDK has a help library that provides help topics for programmers and application developers who work with ArcGIS to write add-ins and custom ArcGIS applications.	GIS developers
Administrator Library provides guidance for configuring and managing ArcGIS installations, geodatabases, and Web-based ArcGIS systems.	ArcGIS administrators, DBAs, and Web architects

About ArcGIS

ArcGIS provides a scalable framework for implementing GIS for a single user or many users on desktops, in servers, over the Web, and in the field. ArcGIS is an integrated family of GIS software products for building a complete GIS. It consists of several primary frameworks for deploying GIS:

- **ArcGIS Desktop**—An integrated suite of professional GIS applications. Most users recognize this as three products: ArcView, ArcEditor, and ArcInfo.
- **ArcGIS Server**—Publishes GIS information and maps as Web services, provides a series of Web GIS applications, and supports enterprise data management.
- **ArcGIS Mobile**—Provides mobile GIS tools and applications for field computing.

Pick any ArcGIS tool - help is not helpful



Tool Help is much better

Contents Favorites Search

- Welcome to ArcGIS Help
- What's new in ArcGIS
- Essentials Library
- Professional Library
- Administrator Library
- GIS glossary
- Copyright information
- License agreement
- Acknowledgements

Cell Statistics (Spatial Analyst)

ArcGIS 10

Summary

Calculates a per-cell statistic from multiple rasters.

The available statistics are Majority, Maximum, Mean, Median, Minimum, Minority, Range, Standard Deviation, Sum, and Variety.

[Learn more about how Cell Statistics works](#)

Illustration

`OutRas = CellStatistics([InRas1, InRas2, InRas3], "SUM", "DATA")`

Usage

- The order of the input rasters is irrelevant for this tool.
- For statistic types Maximum, Minimum, Mean, Median, Majority, Minority and Sum, if a single raster is used as the input, the output cell values will be the same as the input cell values. For Range and STD the output cell values will all be 0, and for Variety, all 1.

Syntax

`CellStatistics (in_rasters_or_constants, {statistics_type}, {ignore_nodata})`

Parameter	Explanation	Data Type
<code>in_rasters_or_constants</code> [<code>in_raster_or_constant,...</code>]	A list of input rasters for which a statistic will be calculated for each cell within the Analysis window. A number can be used as an input; however, the cell size and extent must first be set in the environment.	Raster Layer Constant
<code>statistics_type</code> (Optional)	Statistic type to be calculated. <ul style="list-style-type: none">MEAN — Calculates the mean (average) of the inputs.MAJORITY — Determines the majority (value that occurs most often) of the inputs.	String

Scroll to bottom for sample Python script

CellStatistics example (stand-alone script)

This example calculates the standard deviation per cell on several input GRID rasters and outputs the result as a GRID raster.

```
# Name: CellStatistics_Ex_02.py
# Description: Calculates a per-cell statistic from multiple rasters
# Requirements: Spatial Analyst Extension
# Author: ESRI

# Import system modules
import arcpy
from arcpy import env
from arcpy.sa import *

# Set environment settings
env.workspace = "C:/sapyexamples/data"

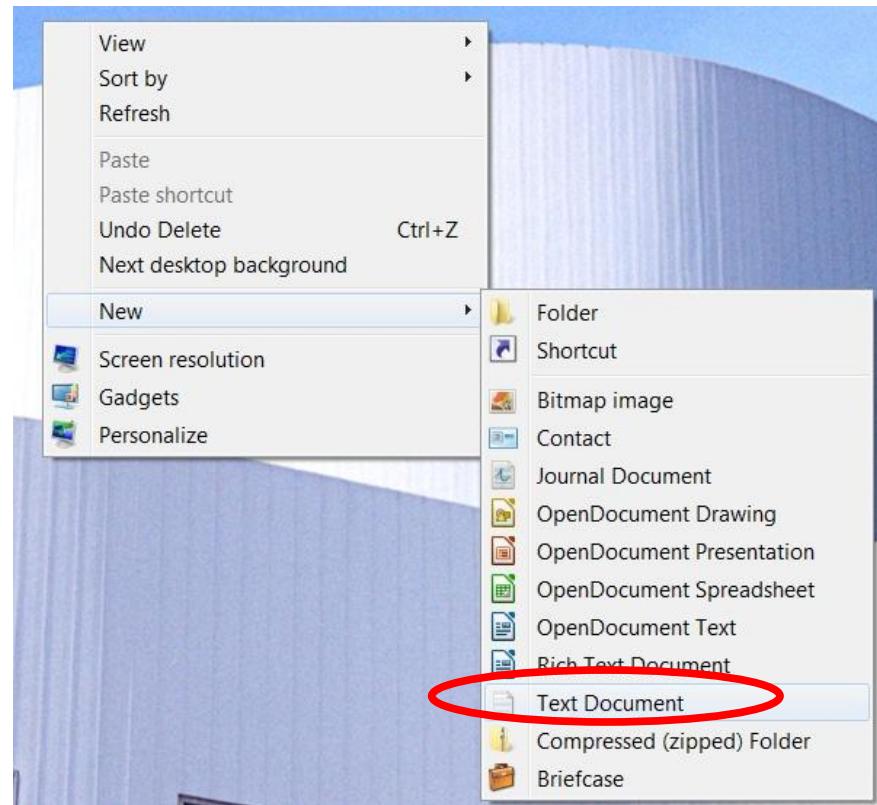
# Set local variables
inRaster01 = "degs"
inRaster02 = "negs"
inRaster03 = "cost"

# Check out the ArcGIS Spatial Analyst extension license
arcpy.CheckOutExtension("Spatial")

# Execute CellStatistics
outCellStatistics = CellStatistics([inRaster01, inRaster02, inRaster03], "RANGE", "NODATA")

# Save the output
outCellStatistics.save("C:/sapyexamples/output/cellstats")
```

Python scripts are text files with “.py” extension



Create new document, copy / paste text

A screenshot of a Windows Notepad window titled "New Text Document.txt - Notepad". The window contains a Python script. The first few lines of the script are:

```
# Name: CellStatistics_Ex_02.py
# Description: Calculates a per cell statistic from multiple rasters
# Requirements: Spatial Analyst Extension
# Author: ESRI
```

The entire first section of the script, which includes the header comments, is circled in red.

```
# Import system modules
import arcpy
from arcpy import env
from arcpy.sa import *

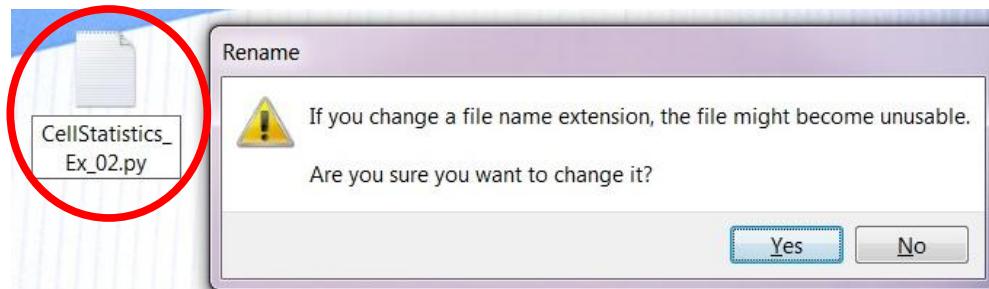
# Set environment settings
env.workspace = "C:/sapyexamples/data"

# Set local variables
inRaster01 = "degs"
inRaster02 = "negs"
inRaster03 = "cost"

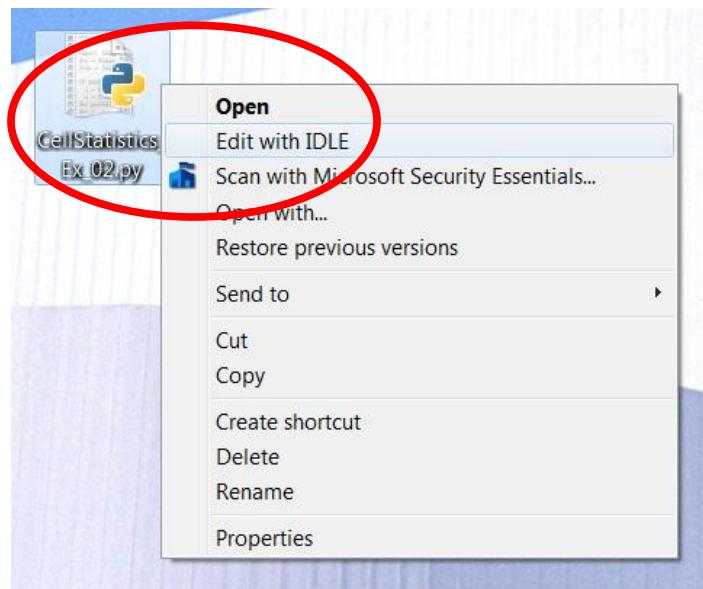
# Check out the ArcGIS Spatial Analyst extension license
arcpy.CheckOutExtension("Spatial")

# Execute CellStatistics
outCellStatistics = CellStatistics([inRaster01, inRaster02, inRaster03], "RANGE", "NODATA")

# Save the output
outCellStatistics.save("C:/sapyexamples/output/cellstats")
```



For existing scripts, right click and edit with IDLE



IDLE displays color-coded script (Don't need the Python Shell, close it)

The screenshot shows the Python IDLE interface. At the top, there's a menu bar with File, Edit, Debug, Options, Windows, and Help. Below the menu bar, the status bar displays "Python 2.6.5 (r265:79096, Mar 19 2010, 21:48:26) [MSC v.1500 32 bit (Intel)] on win32". The main area contains a code editor with a red oval highlighting the menu bar of the editor window. The code in the editor is as follows:

```
*****  
Personal fire makes to its interface. This interface and  
*****  
# Import system modules  
import arcpy  
from arcpy import env  
from arcpy.sa import *  
  
# Set environment settings  
env.workspace = "C:/sapyexamples/data"  
  
# Set local variables  
inRaster01 = "degs"  
inRaster02 = "negs"  
inRaster03 = "cost"  
  
# Check out the ArcGIS Spatial Analyst extension license  
arcpy.CheckOutExtension("Spatial")  
  
# Execute CellStatistics  
outCellStatistics = CellStatistics([inRaster01, inRaster02, inRaster03], "RANGE", "NODATA")  
  
# Save the output  
outCellStatistics.save("C:/sapyexamples/output/cellstats")
```

Example

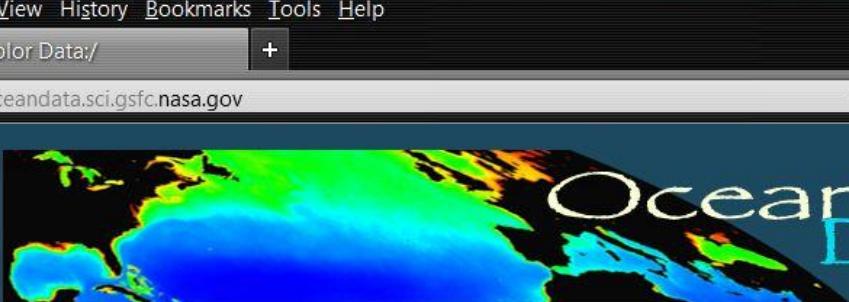
Get GIS data

OceanColor Data:/ - Mozilla Firefox

File Edit View History Bookmarks Tools Help

OceanColor Data:/ +

oceanodata.sci.gsfc.nasa.gov arcgis 10 global statistics e

 OceanColor Data

Missions Data Documents Analyses People Forum Services Links

Google Custom Search Search

Data Distribution Site Description

In addition to browsing the directory structure below, you may create a Customizable File Search

Filename	Last Modified	Size
Ancillary	-	-
CZCS	-	-
MERIS	-	-
MODISA	-	-
MODIST	-	-
OCTS	-	-
SeaWiFS	-	-
VIIRS	-	-
Aquarius	-	-

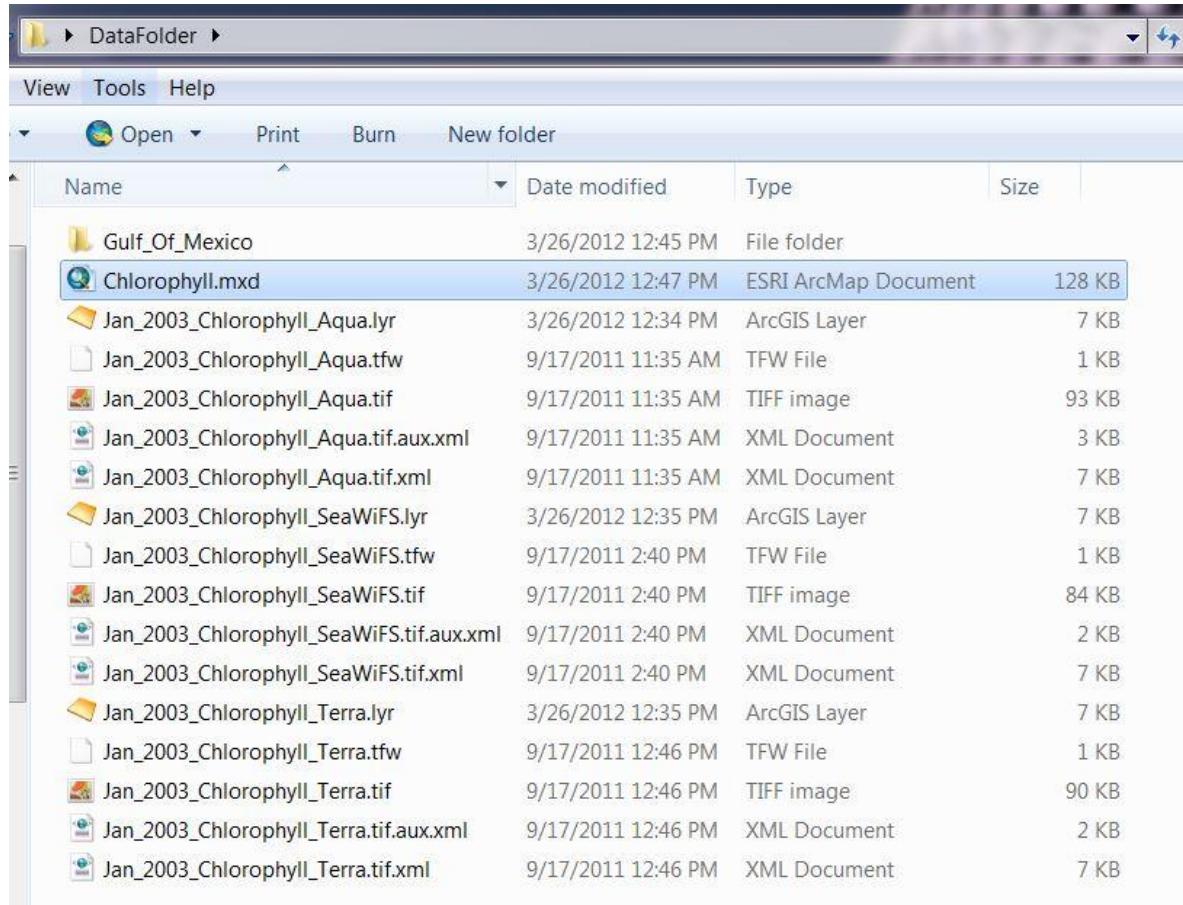
Curator: OceanColor Webmaster Privacy Policy and Important Notices

Authorized by: gene carl feldman Updated: 10 June 2011

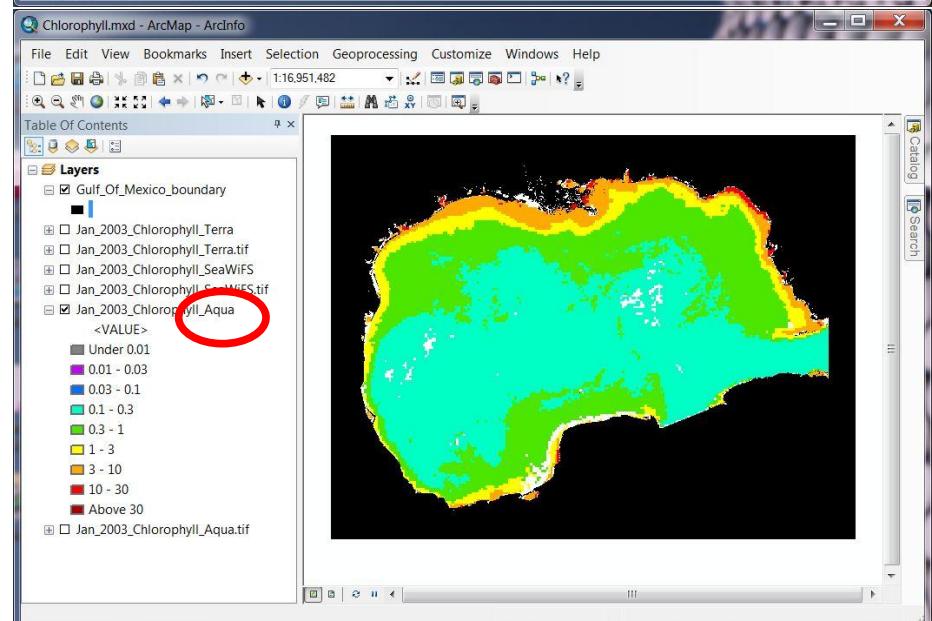
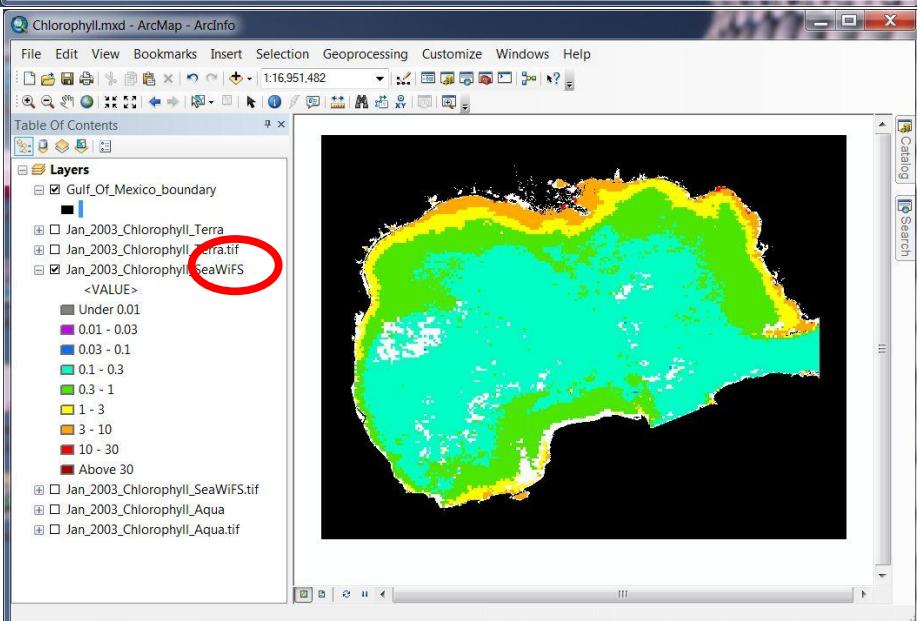
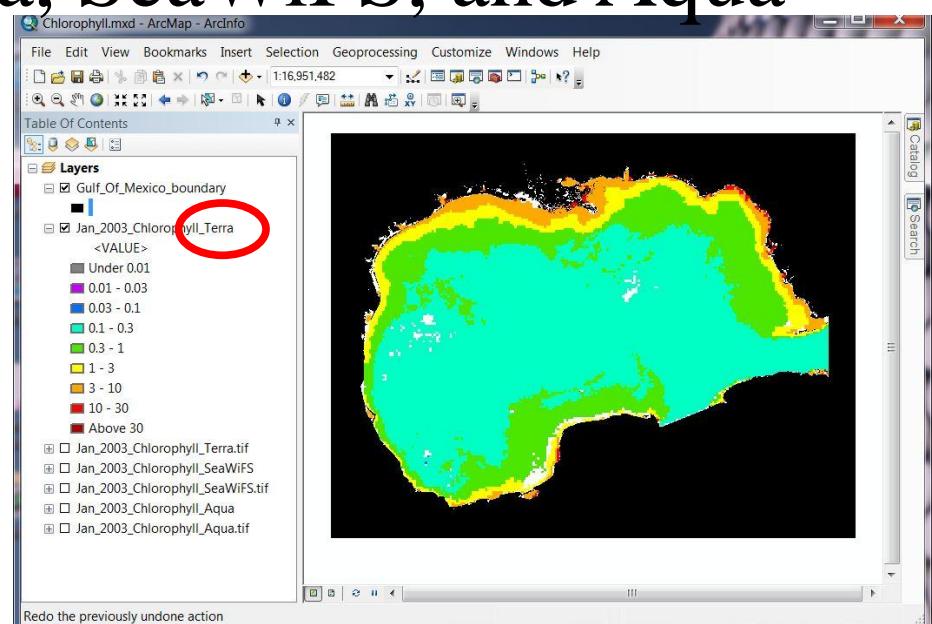
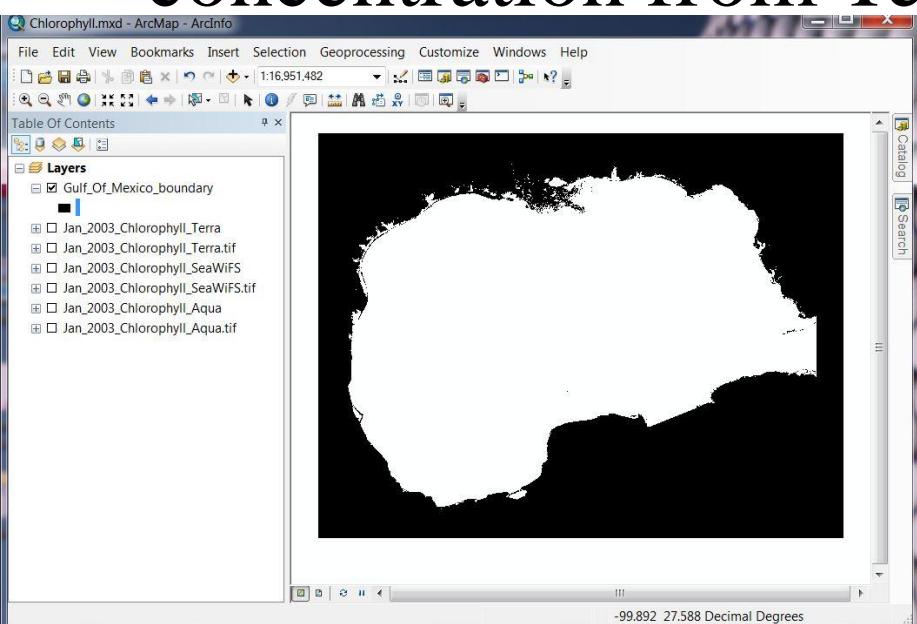
NASA

Example

Downloaded, processed, and symbolized layers



Gulf Of Mexico outline, plus monthly Chlorophyll concentration from Terra, SeaWiFS, and Aqua



Edit script to match data location and names

```
Mean.py - C:\Users\Kagome\Desktop\Mean.py
File Edit Format Run Options Windows Help
# Name: Mean.py
# Description: Calculates mean chlorophyll concentration from 3 input rasters
# Requirements: Spatial Analyst Extension
# Author: Name

# Import system modules
import arcpy
from arcpy import env
from arcpy.sa import *

# Set environment settings
env.workspace = "C:/Users/Kagome/Desktop/DataFolder"

# Set local variables
inRaster01 = "Jan_2003_Chlorophyll_Terra.tif"
inRaster02 = "Jan_2003_Chlorophyll_SeaWiFS.tif"
inRaster03 = "Jan_2003_Chlorophyll_Aqua.tif"
outRaster = "Jan_2003_Chlorophyll_Mean.tif"

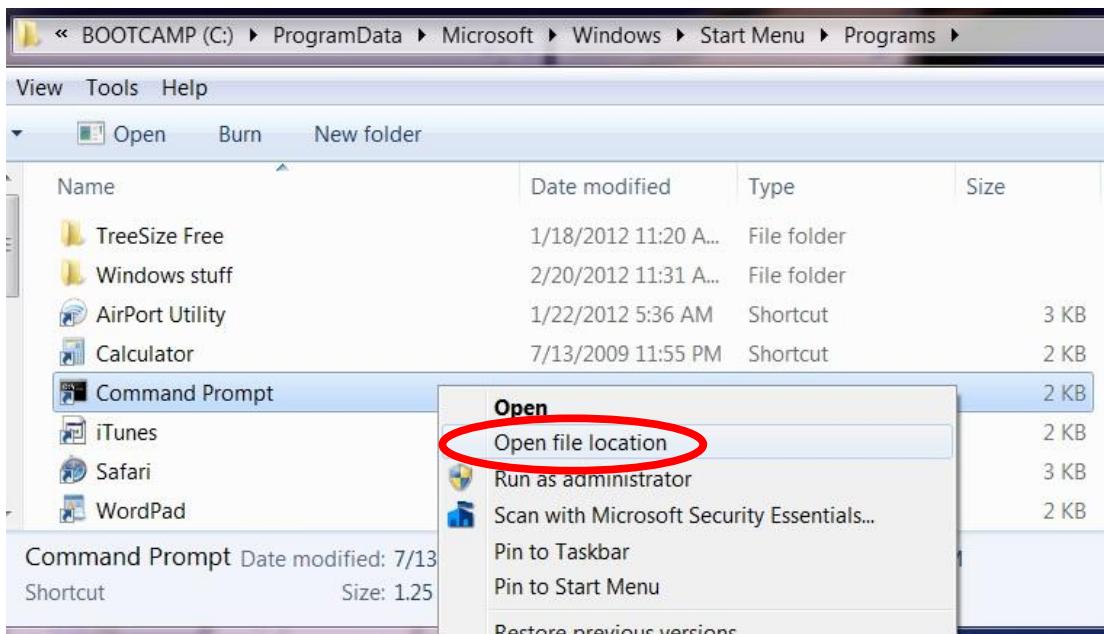
# Check out the ArcGIS Spatial Analyst extension license
arcpy.CheckOutExtension("Spatial")

# Execute CellStatistics
outCellStatistics = CellStatistics([inRaster01, inRaster02, inRaster03], "MEAN", "NODATA")

# Save the output
outCellStatistics.save(outRaster)
```

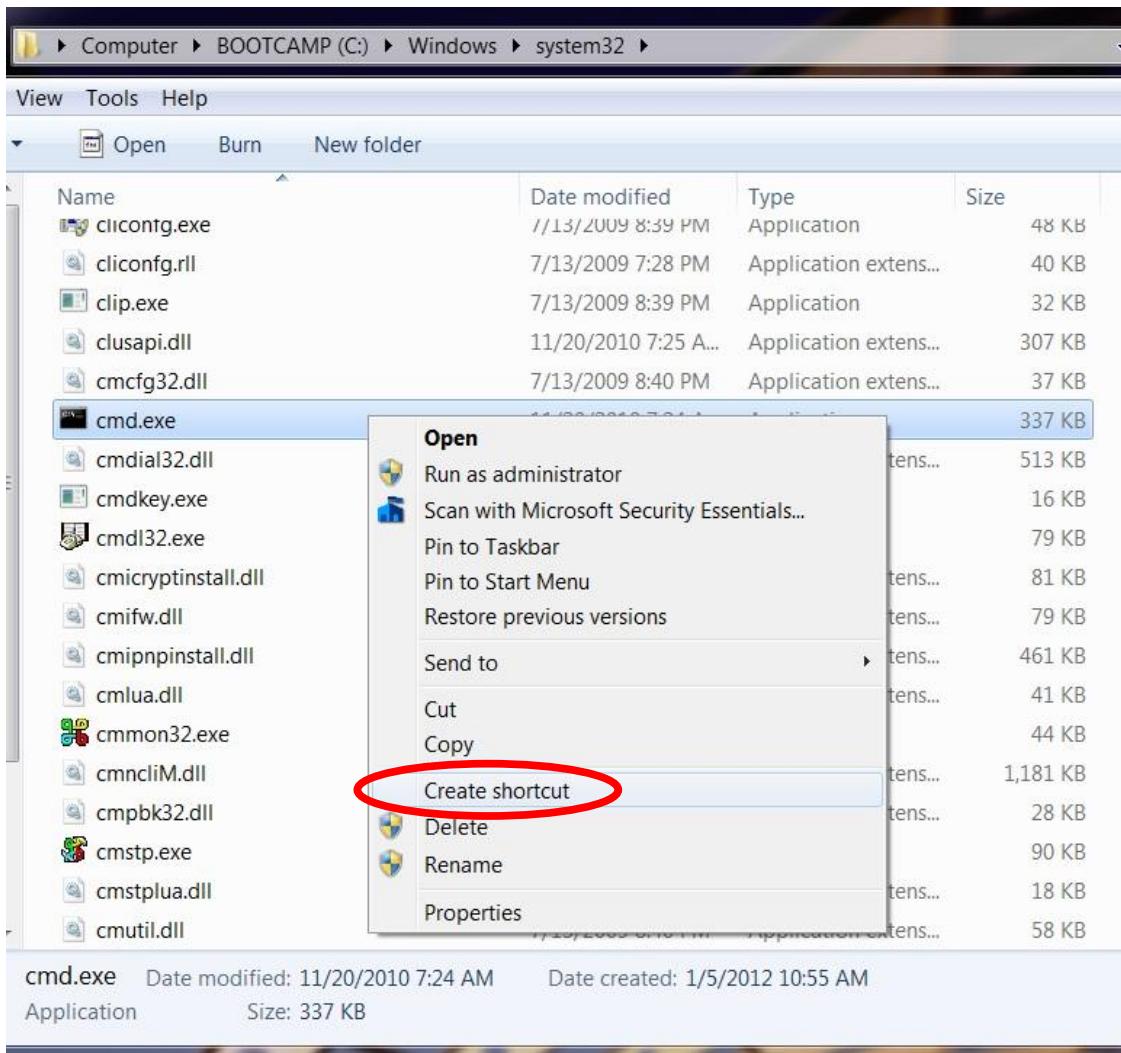
Setup

Run Python from Windows Command Prompt



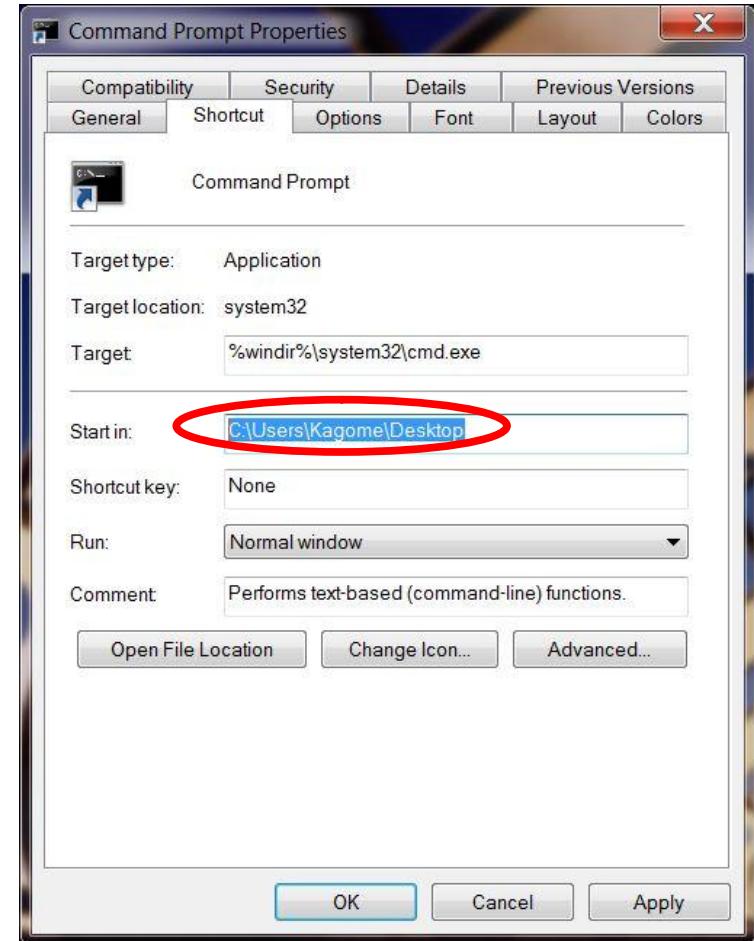
Setup

Easier to create a customized shortcut

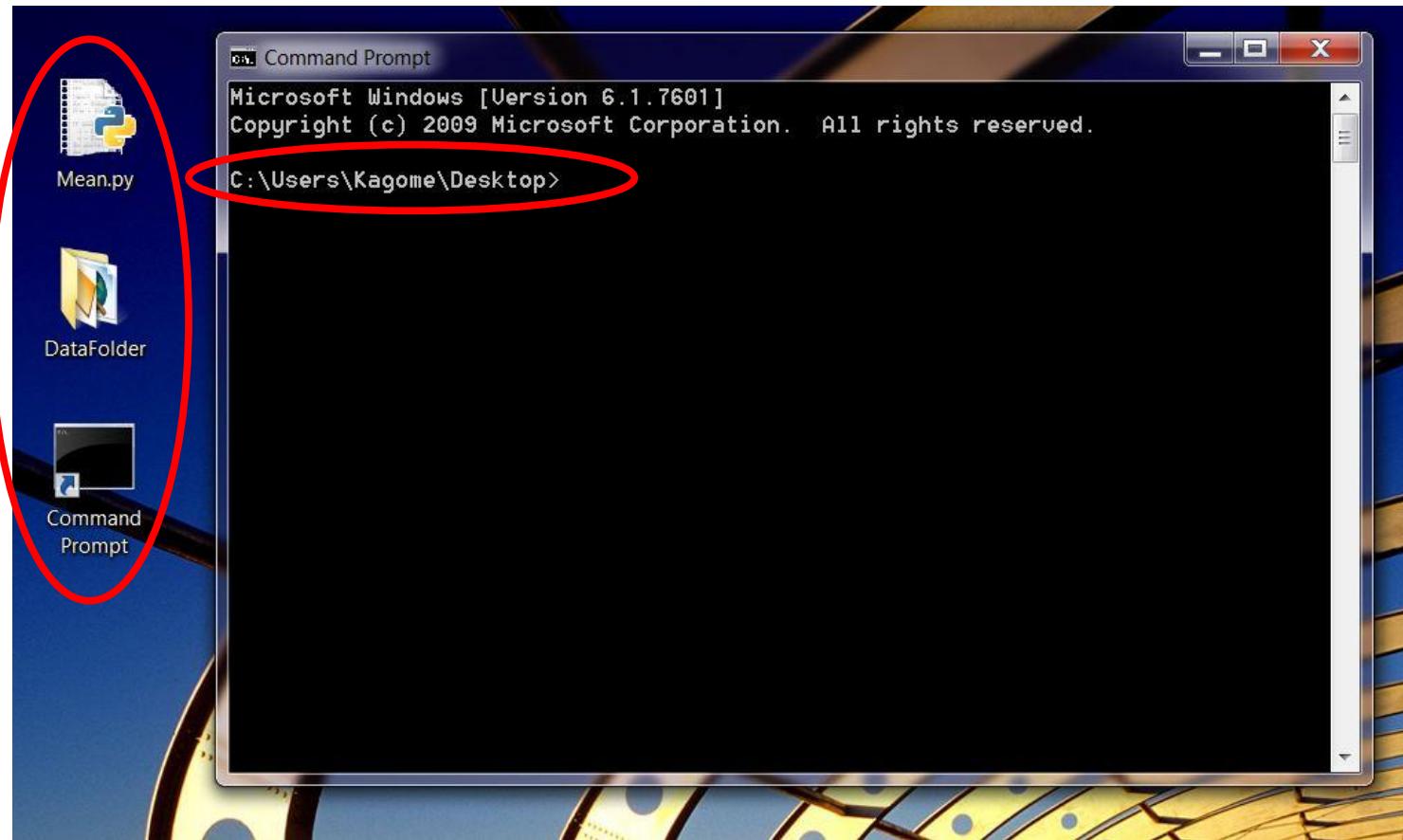


Setup

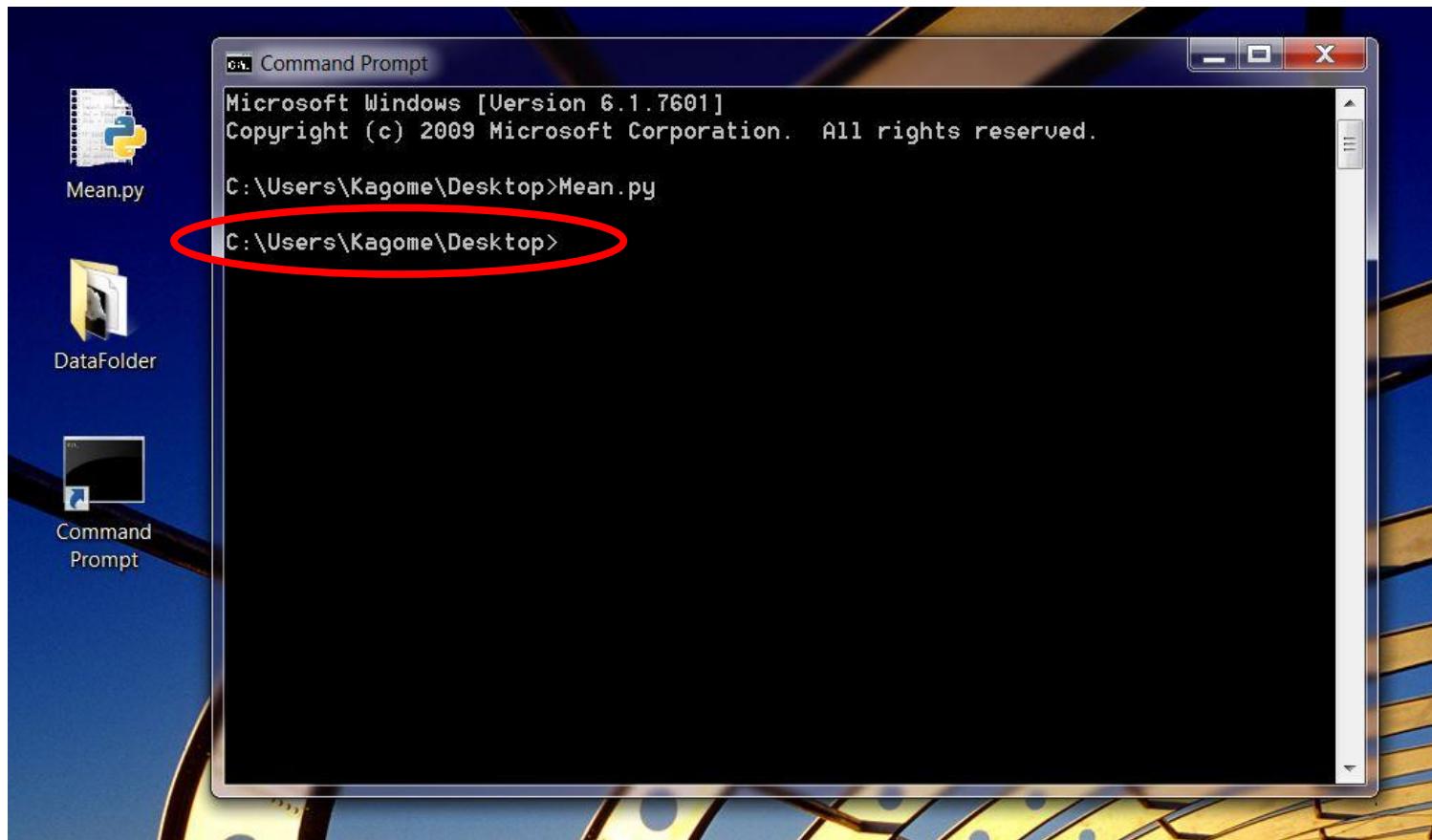
Start Command Prompt in same folder as script



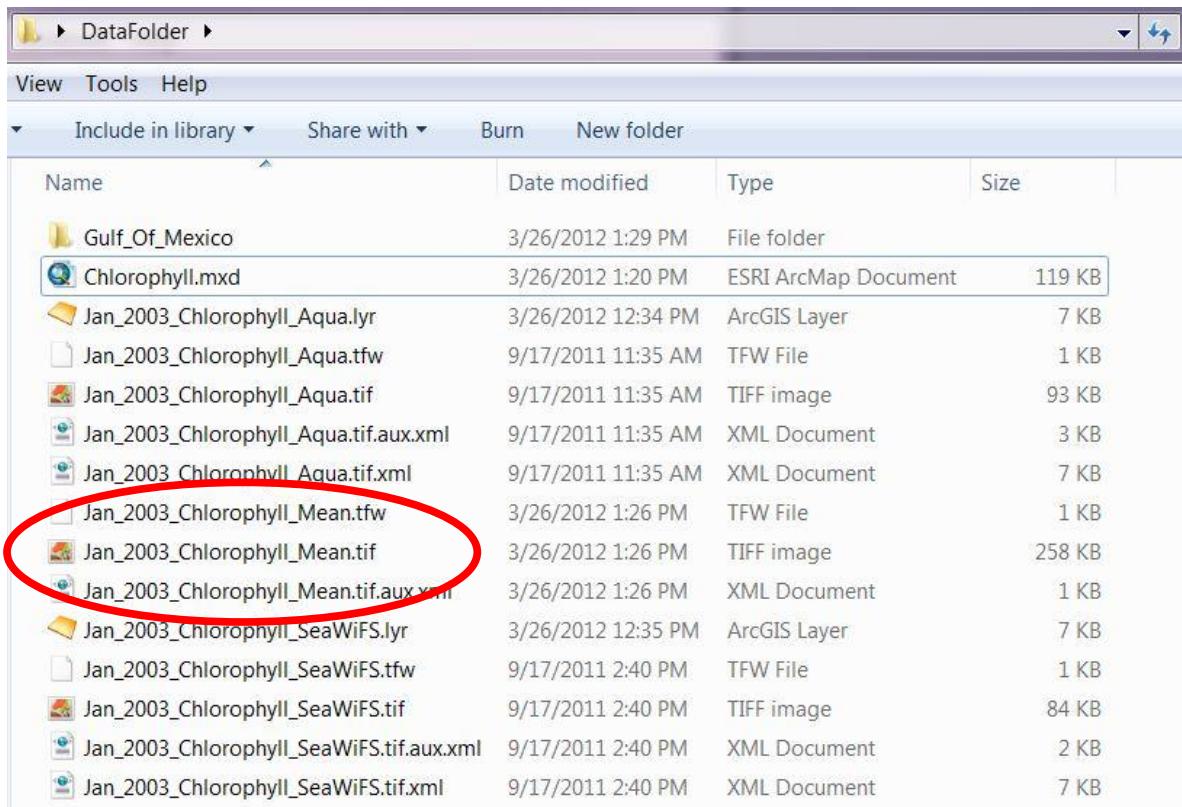
Desktop and Command Prompt ready



Run... no feedback (maybe you like that)

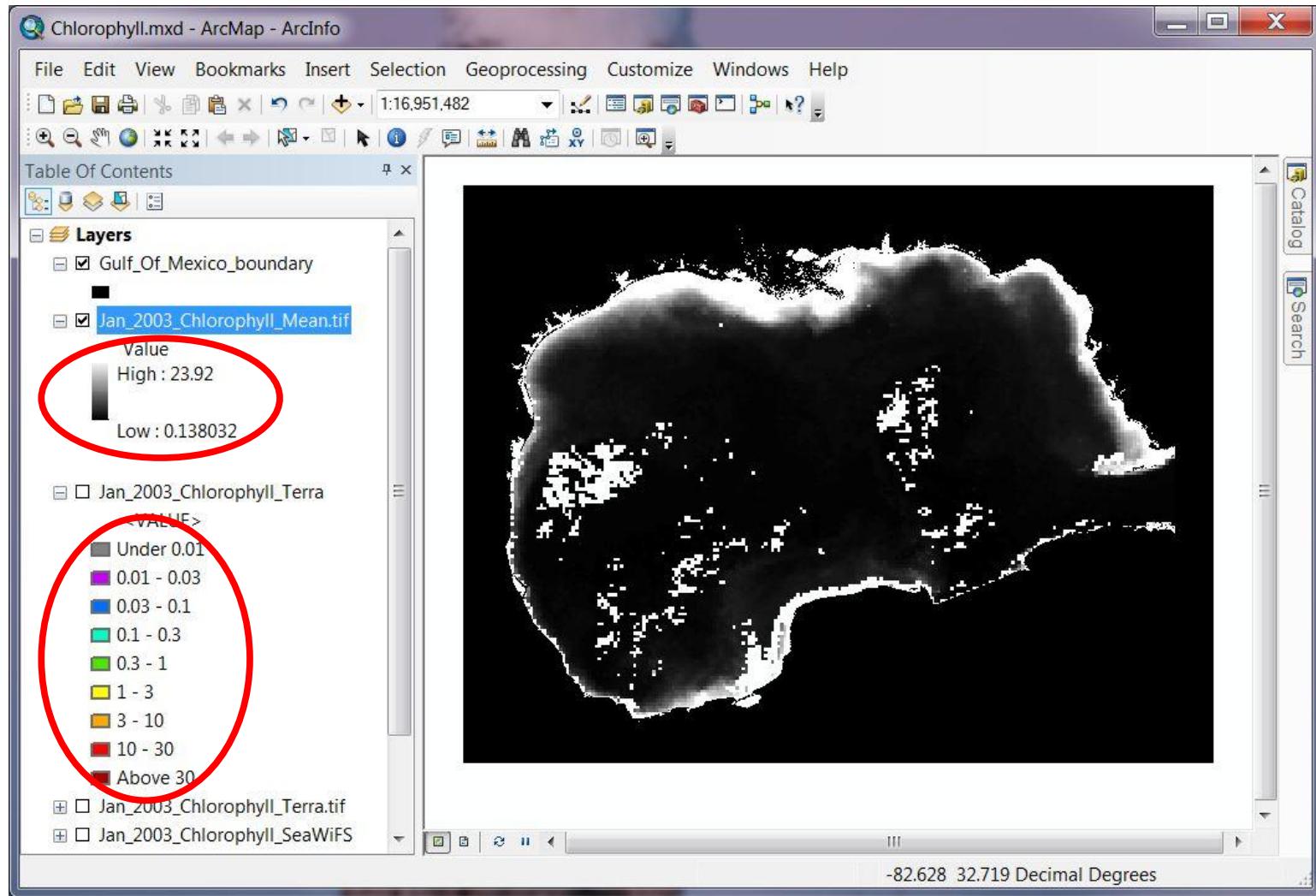


Script created “Mean” datafiles

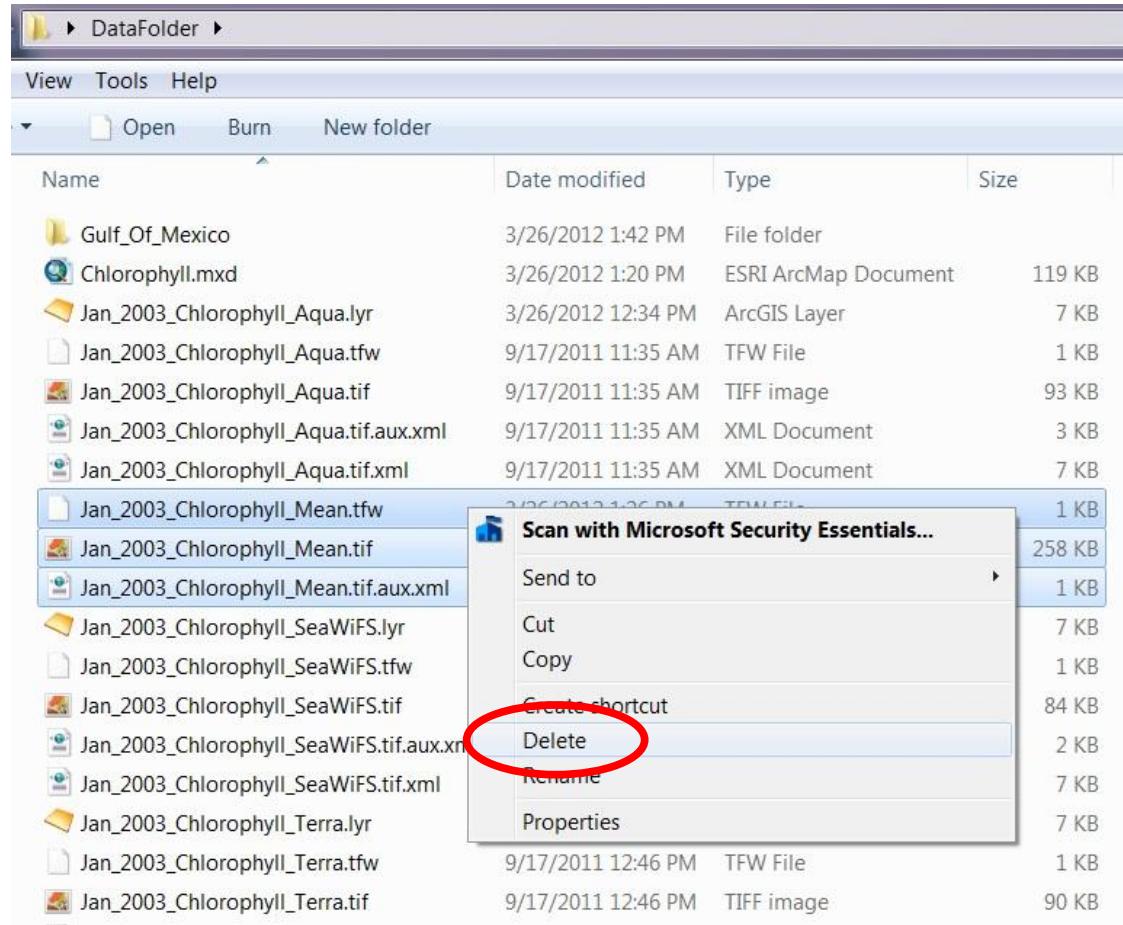


Name	Date modified	Type	Size
Gulf_Of_Mexico	3/26/2012 1:29 PM	File folder	
Chlorophyll.mxd	3/26/2012 1:20 PM	ESRI ArcMap Document	119 KB
Jan_2003_Chlorophyll_Aqua.lyr	3/26/2012 12:34 PM	ArcGIS Layer	7 KB
Jan_2003_Chlorophyll_Aqua.tfw	9/17/2011 11:35 AM	TFW File	1 KB
Jan_2003_Chlorophyll_Aqua.tif	9/17/2011 11:35 AM	TIFF image	93 KB
Jan_2003_Chlorophyll_Aqua.tif.aux.xml	9/17/2011 11:35 AM	XML Document	3 KB
Jan_2003_Chlorophyll_Aqua.tif.xml	9/17/2011 11:35 AM	XML Document	7 KB
Jan_2003_Chlorophyll_Mean.tfw	3/26/2012 1:26 PM	TFW File	1 KB
Jan_2003_Chlorophyll_Mean.tif	3/26/2012 1:26 PM	TIFF image	258 KB
Jan_2003_Chlorophyll_Mean.tif.aux.xml	3/26/2012 1:26 PM	XML Document	1 KB
Jan_2003_Chlorophyll_SeaWiFS.lyr	3/26/2012 12:35 PM	ArcGIS Layer	7 KB
Jan_2003_Chlorophyll_SeaWiFS.tfw	9/17/2011 2:40 PM	TFW File	1 KB
Jan_2003_Chlorophyll_SeaWiFS.tif	9/17/2011 2:40 PM	TIFF image	84 KB
Jan_2003_Chlorophyll_SeaWiFS.tif.aux.xml	9/17/2011 2:40 PM	XML Document	2 KB
Jan_2003_Chlorophyll_SeaWiFS.tif.xml	9/17/2011 2:40 PM	XML Document	7 KB

Open in ArcMap... unsymbolized



ArcPython does not like to overwrite files



Add status messages, create symbolize layer

The screenshot shows a Windows-style application window titled "Mean.py - C:\Users\Kagome\Desktop\Mean.py". The menu bar includes File, Edit, Format, Run, Options, Windows, and Help. The main code area contains a Python script for calculating the mean chlorophyll concentration from three input rasters. A red rectangular box highlights the section of code that creates an ArcMap layer and applies symbology. The script ends with a status message.

```
# Name: Mean.py
# Description: Calculates mean chlorophyll concentration from 3 input rasters
# Requirements: Spatial Analyst Extension
# Author: Name

# Import system modules
import arcpy
from arcpy import env
from arcpy.sa import *

print "Initializing script"

# Set environment settings
env.workspace = "C:/Users/Kagome/Desktop/DataFolder"

# Set local variables
inRaster01 = "Jan_2003_Chlorophyll_Terra.tif"
inRaster02 = "Jan_2003_Chlorophyll_SeaWiFS.tif"
inRaster03 = "Jan_2003_Chlorophyll_Aqua.tif"
outRaster = "Jan_2003_Chlorophyll_Mean.tif"

# Check out the ArcGIS Spatial Analyst extension license
arcpy.CheckOutExtension("Spatial")

print "Calculating mean"

# Execute CellStatistics
outCellStatistics = CellStatistics([inRaster01, inRaster02, inRaster03], "MEAN", "NODATA")

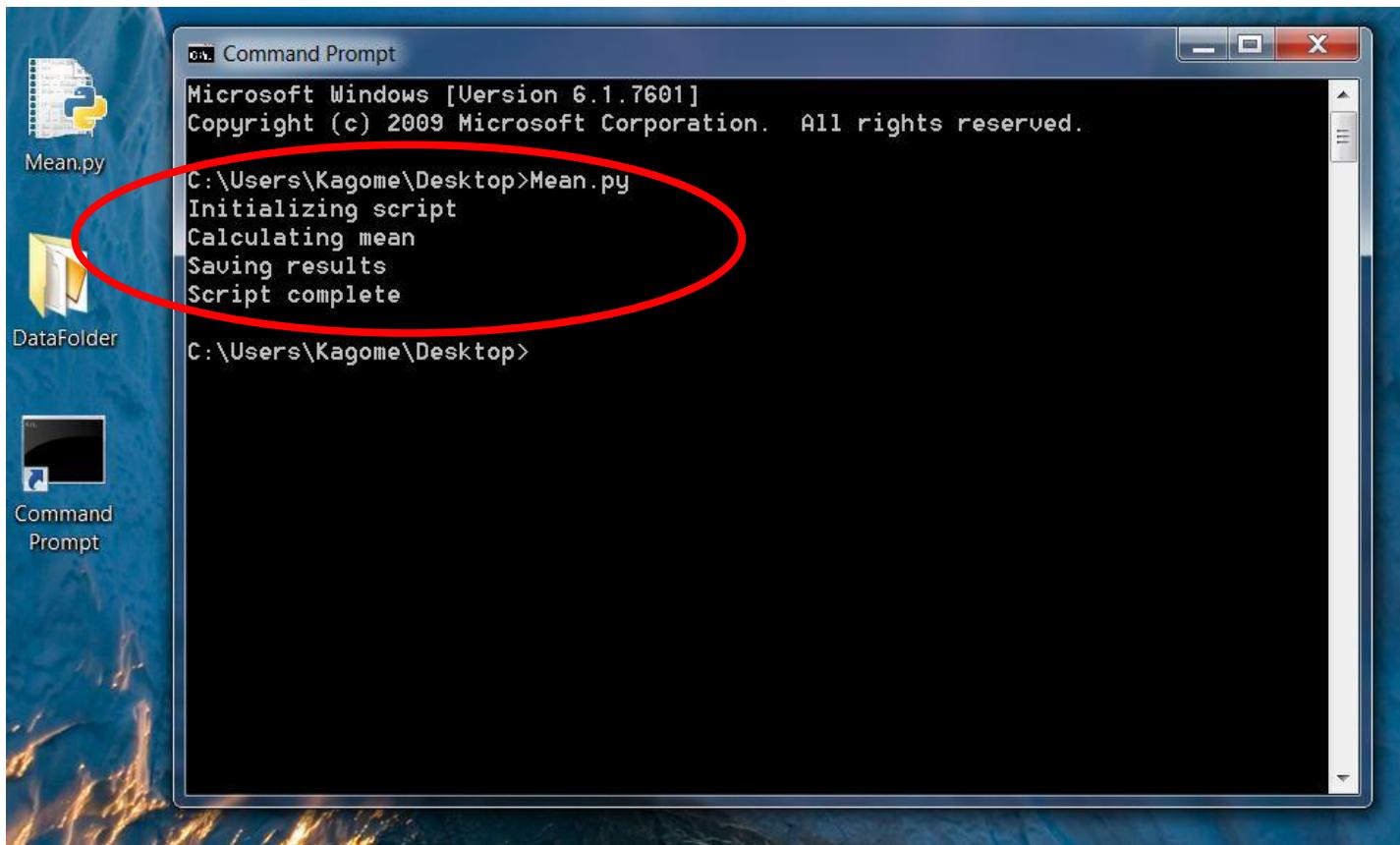
print "Saving results"

# Save the output
outCellStatistics.save(outRaster)

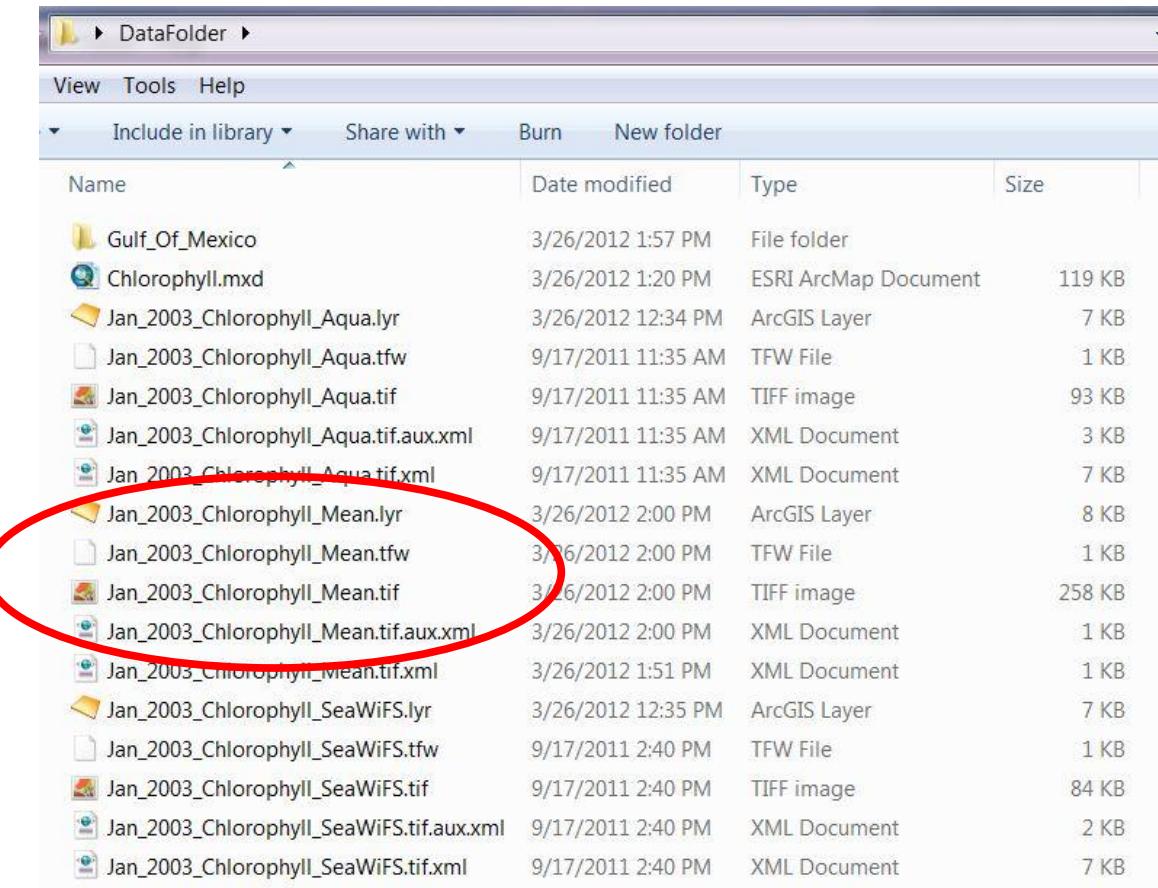
# Make output raster into an ArcMap layer, apply color scheme from first input raster
inRasterLayer = "Jan_2003_Chlorophyll_Terra.lyr"
outLayerName = "Jan_2003_Chlorophyll_Mean"
outLayerFile = "Jan_2003_Chlorophyll_Mean.lyr"
arcpy.MakeRasterLayer_management(outRaster, outLayerName)
arcpy.ApplySymbologyFromLayer_management(outLayerName, inRasterLayer)
arcpy.SaveToLayerFile_management(outLayerName, outLayerFile, "RELATIVE")

print "Script complete"
```

Run again... feedback shows

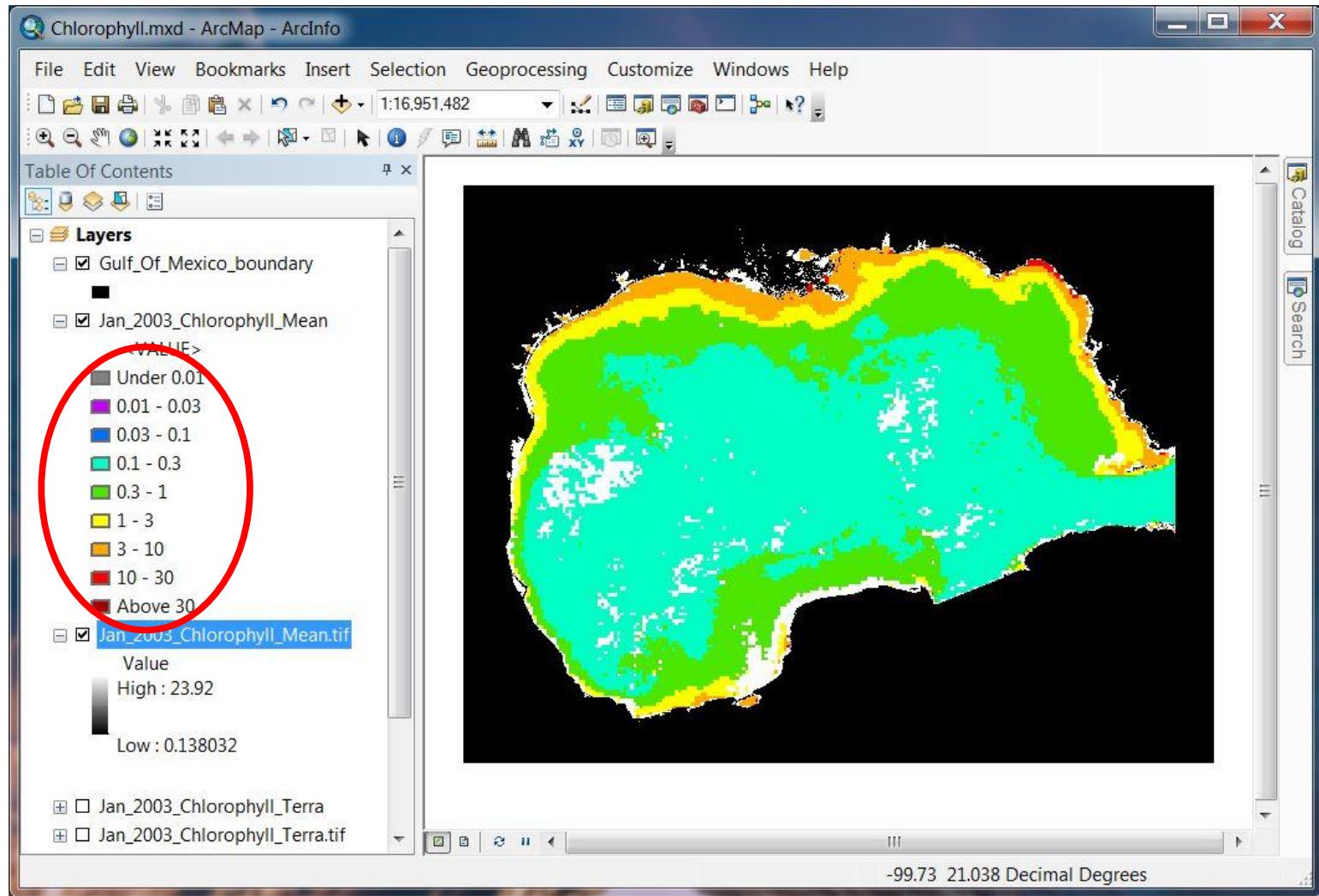


Script created “Mean” datafiles and layer



Name	Date modified	Type	Size
Gulf_Of_Mexico	3/26/2012 1:57 PM	File folder	
Chlorophyll.mxd	3/26/2012 1:20 PM	ESRI ArcMap Document	119 KB
Jan_2003_Chlorophyll_Aqua.lyr	3/26/2012 12:34 PM	ArcGIS Layer	7 KB
Jan_2003_Chlorophyll_Aqua.tfw	9/17/2011 11:35 AM	TFW File	1 KB
Jan_2003_Chlorophyll_Aqua.tif	9/17/2011 11:35 AM	TIFF image	93 KB
Jan_2003_Chlorophyll_Aqua.tif.aux.xml	9/17/2011 11:35 AM	XML Document	3 KB
Jan_2003_Chlorophyll_Aqua.tif.xml	9/17/2011 11:35 AM	XML Document	7 KB
Jan_2003_Chlorophyll_Mean.lyr	3/26/2012 2:00 PM	ArcGIS Layer	8 KB
Jan_2003_Chlorophyll_Mean.tfw	3/26/2012 2:00 PM	TFW File	1 KB
Jan_2003_Chlorophyll_Mean.tif	3/26/2012 2:00 PM	TIFF image	258 KB
Jan_2003_Chlorophyll_Mean.tif.aux.xml	3/26/2012 2:00 PM	XML Document	1 KB
Jan_2003_Chlorophyll_Mean.tif.xml	3/26/2012 1:51 PM	XML Document	1 KB
Jan_2003_Chlorophyll_SeaWiFS.lyr	3/26/2012 12:35 PM	ArcGIS Layer	7 KB
Jan_2003_Chlorophyll_SeaWiFS.tfw	9/17/2011 2:40 PM	TFW File	1 KB
Jan_2003_Chlorophyll_SeaWiFS.tif	9/17/2011 2:40 PM	TIFF image	84 KB
Jan_2003_Chlorophyll_SeaWiFS.tif.aux.xml	9/17/2011 2:40 PM	XML Document	2 KB
Jan_2003_Chlorophyll_SeaWiFS.tif.xml	9/17/2011 2:40 PM	XML Document	7 KB

Open in ArcMap... symbolized



Python and ArcGIS are powerful tools

Do not be fooled by simple problems described
and solved by scripts in beginner textbooks

Try manually processing multiple GB of data with
dozens of processing steps, **EVERY DAILY**