2014 INTERNATIONAL USER CONFERENCE
VANCOUVER, B.C. CANADA

Python and FME

FME UC 2014 Training Session

CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Introduction

- Employed at con terra GmbH since 2009
- Spatial ETL Professional Services
- FME Certified Trainer since 2010
- FME Certified Professional since 2013

CERTIFIED Professional

CERTIFIED Trainer

CONNECT. TRANSFORM. AUTOMATE.

Agenda

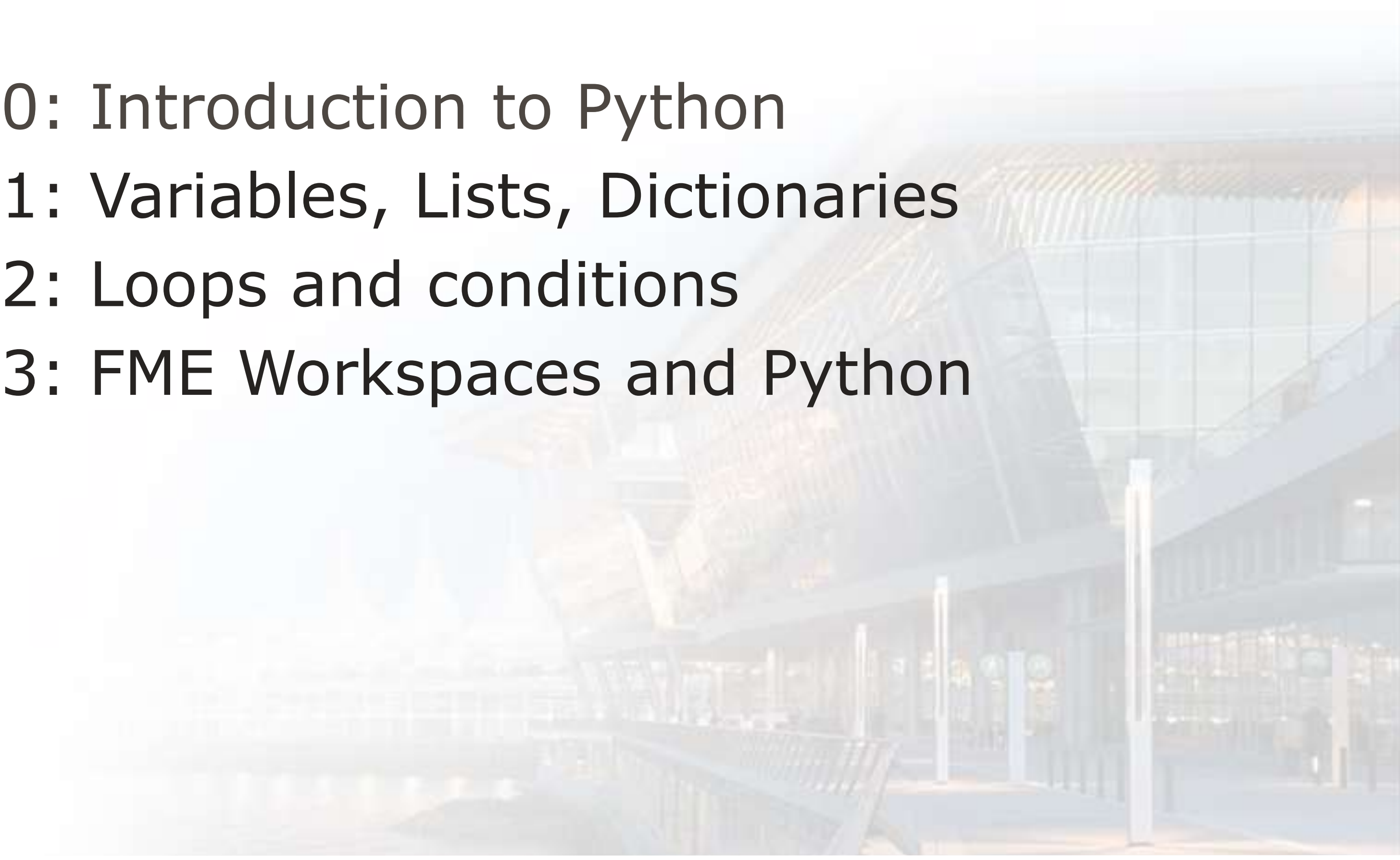
- Introduction
- 4 Timeslots
  - 9am - 10.30am
  - 10.45am - 12.15pm
  - 1.15pm - 2.45pm
  - 3pm - 4.30pm
- Course Content
- Work environment

Resources

- http://fme.ly/fmeucdns
- http://www.safe.com/uctraining

Course content forenoon

- Chapter 0: Introduction to Python
- Chapter 1: Variables, Lists, Dictionaries
- Chapter 2: Loops and conditions
- Chapter 3: FME Workspaces and Python



Course content afternoon

- Chapter 4: Configuring Eclipse
- Chapter 5: Python WorkspaceRunner
- Chapter 6: Custom Format Reader
- Chapter 7: Debugging

Hardware

- Your computer
- Printed exercise manual
- Personal EC2 image

2014 International User Conference
Vancouver, B.C. Canada

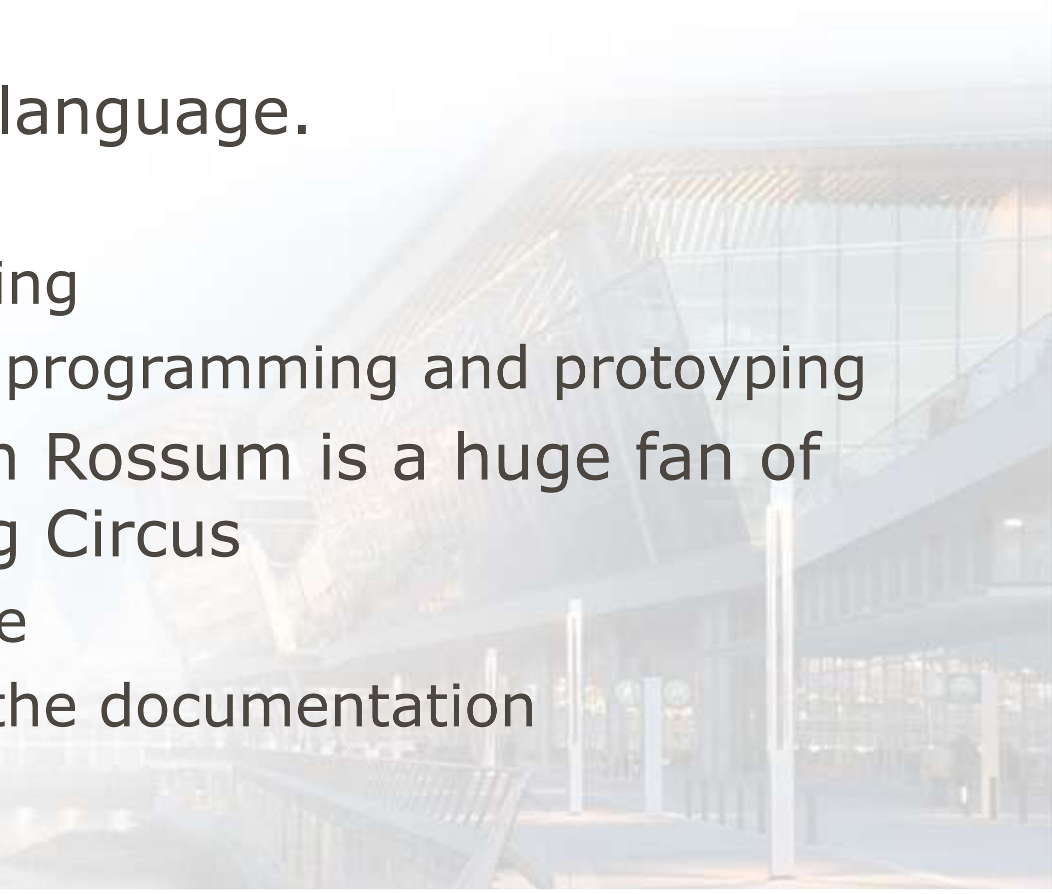
Chapter 0

- Introduction to Python

CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Python



- Python is a scripting language.
  - But Object oriented
  - No compiling or linking
  - Fast (“quick&dirty”) programming and protoyping
- Name: Developer van Rossum is a huge fan of Monty Python’s Flying Circus
  - IDE IDLE=> Eric Idle
  - Many references in the documentation

Why Python?

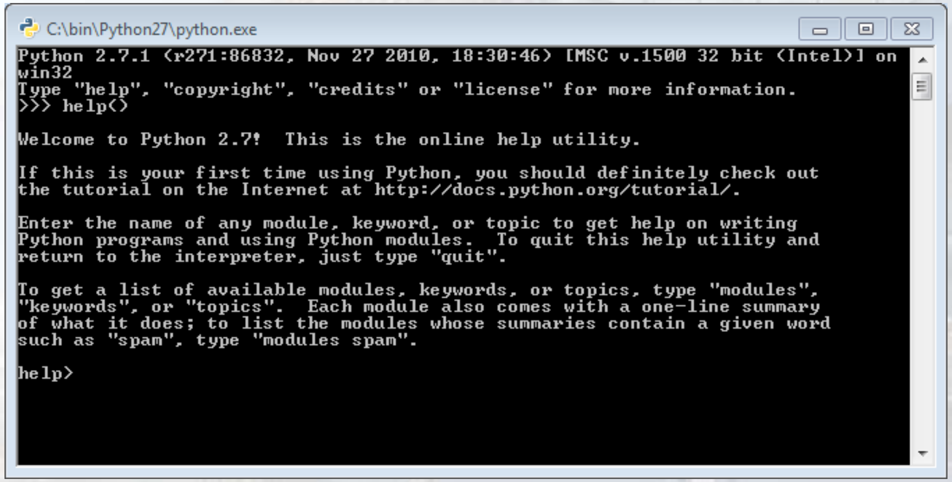
- Free, powerful and flexible
- Platform independent
- Automatic Garbage Collecting
- Capable of being integrated
  - e.g. FME, ArcGIS, Blender
- Extensive documentation
  - www.python.org

Python version havoc

- Latest versions - Python 2.7.7 und Python 3.4.1
- Python 3.x and 2.x are incompatible
  - Most libraries available for both versions
  - 2.x support and (security) bugfixes until 2020
- FME 2014 supports Python 2.7 (- 2.5)
- More details on www.python.org

Interactive Shell

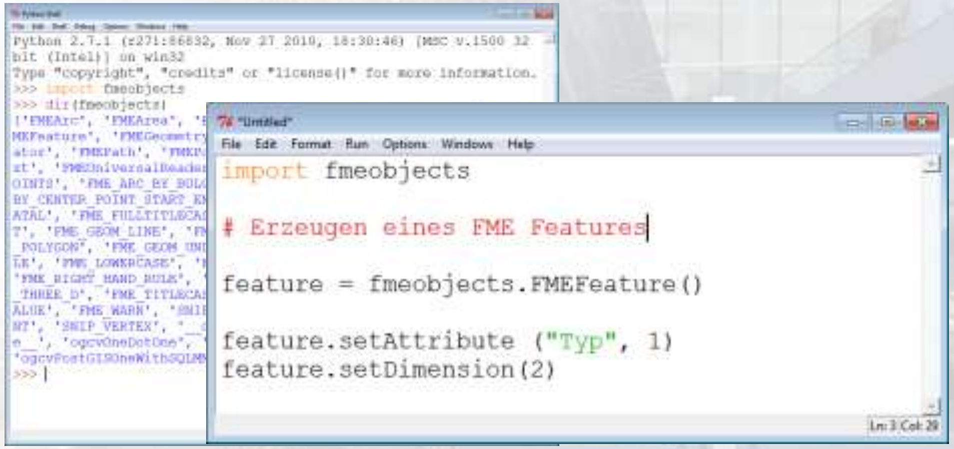
- *Start > Programs > Python 2.7 > Python (command line)*
- Execute single statements
- Not very comfortable



```
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 2.7!  This is the online help utility.

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, or topics, type "modules",
"keywords", or "topics".  Each module also comes with a one-line summary
of what it does; to list the modules whose summaries contain a given word
such as "spam", type "modules spam".

help>
```

Python IDE – Integrated Development Environment

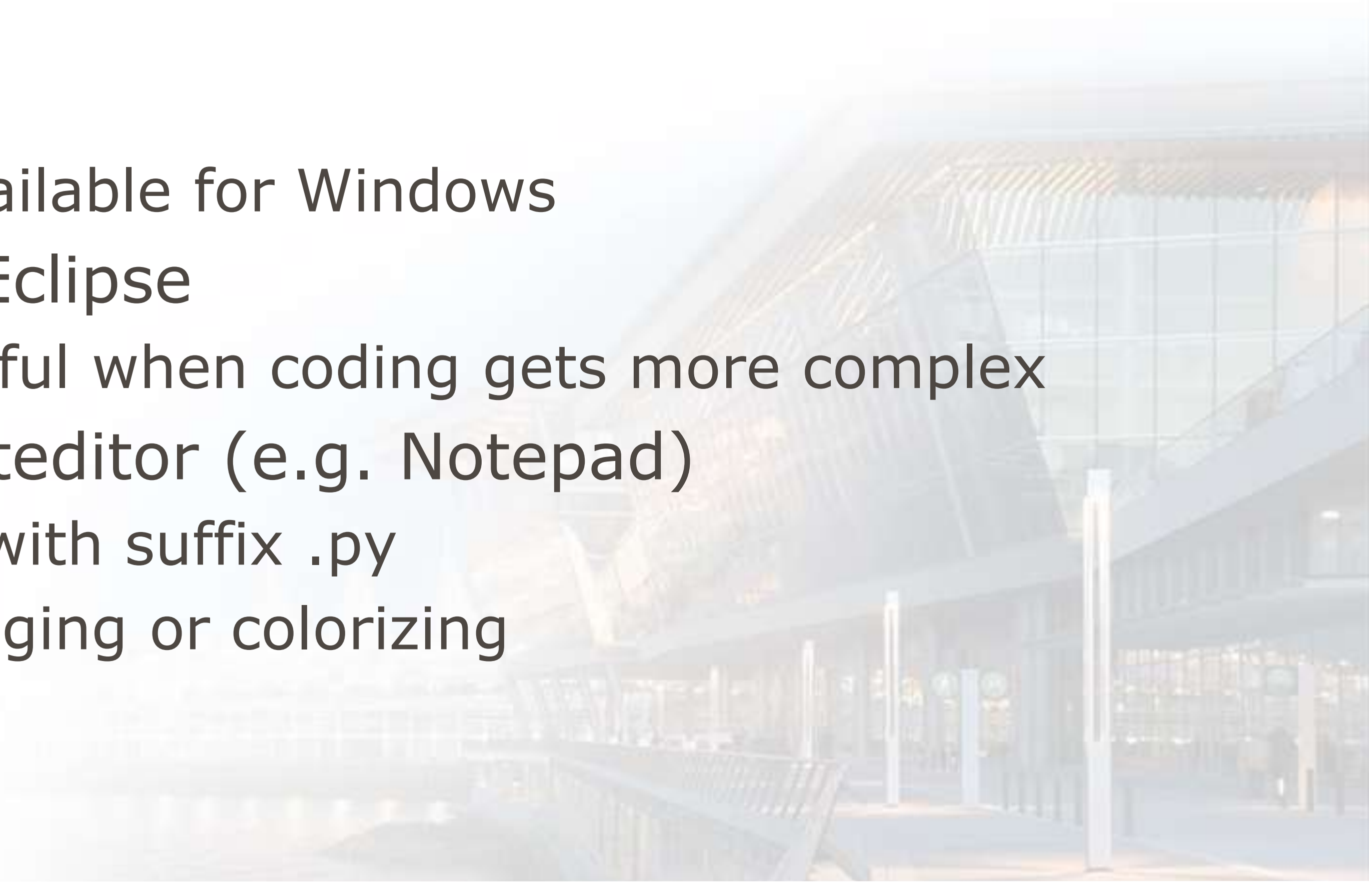
- IDLE – Shell with advanced functionality:
  - Included in the Python default distribution package
  - Interactive- and Script window
  - Code completion
  - Colorizing
  - Debugging
  - Call tips


Python Shell
File Edit Shell Debug Options Windows Help
Untitled
File Edit Format Run Options Windows Help
Ln: 3 Col: 29

```
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import fmeobjects
>>> dir(fmeobjects)
['FMEArc', 'FMEArea', '
MKFeature', 'FMEGeometry
ator', 'FMEPath', 'FMEP
rt', 'FMEUniversalReader
OINTS', 'FME_ARC_BY_BULG
BY_CENTER_POINT_START_EN
ATAL', 'FME_FULLTITLECAS
T', 'FME_GEOM_LINE', 'FM
_POLYGON', 'FME_GEOM_UND
LE', 'FME_LOWERCASE', 'F
'FME_RIGHT_HAND_RULE', '
_THREE_D', 'FME_TITLECAS
ALUE', 'FME_WARN', 'SNIP
NT', 'SNIP_VERTEX', '__
e__', 'ogcvOneDotOne', '
'ogcvPostGISOneWithSQLMM
>>> |
```

```python
import fmeobjects

# Erzeugen eines FME Features

feature = fmeobjects.FMEFeature()

feature.setAttribute ("Typ", 1)
feature.setDimension(2)
```

More IDEs



- PythonWin
  - Editor available for Windows
- PyDev for Eclipse
  - Very helpful when coding gets more complex
- Simple Texteditor (e.g. Notepad)
  - Safe file with suffix .py
  - No debugging or colorizing
  - No shell

Interactive mode

- One line = one statement
- Exceptions:
  - Separate several commands in one line with semicolon
  - One command over several lines:
    - With Backslash ("\")
    - Strings with triple quotation marks
    - Using brackets

```
>>> a = 1+2; print a
3
>>> a = 3 + \
   4
>>> print a
7
```

```
>>> print """One String
over two lines!"""
One String
over two lines!
```

```
>>> a = (10 +
    20)
>>> print a
30
```

Python preface

- Case-sensitive
- Variable and module names have to begin with a alphabetic character
  - Any character, numbers or underscores "_" can follow
  - No special characters (e.g. / \ § $ % &)
- You can use both single (‛) and double (‟) quotes
- Comments with # or “““*A comment*””” (triple quotes)
- dir(object) shows all properties of an object

Reserved keywords

- and
- assert
- break
- class
- continue
- def
- del
- elif
- else
- except
- exec
- finally
- for
- from
- global
- if
- import
- In
- is
- lambda
- not
- or
- pass
- print
- raise
- return
- try
- while
- yield

Exercise 1.1 and 1.2

- Get to know the Python commandline and IDLE

2014 International User Conference
Vancouver, B.C.
Canada

Chapter 1

- Variables in Python

CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content forenoon

- Chapter 0: Introduction to Python
- Chapter 1: Variables, Lists, Dictionaries
- Chapter 2: Loops and conditions
- Chapter 3: FME Workspaces and Python

Variables

- Container for everything you want to reuse
- Have a name
- Point to an address in your computer`s memory
- Access the value via the name
- No declaration in Python
    - Dynamic typing

```
a = 31     # Variable „a“ is a number (Integer)
name = „Don“     # Variable „name“ is a String
feature = fmeobjects.FMEFeature() # Variable „Feature“ ist an object
```

Numbers

- Integer, Long, Double…
- Basic arithmetics +, -, *, /
- Modulo-Operator %
- Exponent **

```
>>> 3/2
1
>>> 3.0 / 2
1.5
```

```
>>> 3+3
6
>>> 3-3
0
>>> 3*3
9
>>> a=3.0;b=3.0
>>> print a/b
1.0
#Exponent
>>> 3**3
27
#Modulo operator
>>> 13%5
3
```

More on numbers

- Many mathematical functions available via module math.
  - A default module, but has to be imported
- Examples
  - > Square root
  - > Constants, e.g. Pi
- Show properties of math:
  - > dir(math)

```
>>> import math
>>> math.sqrt(2)
1.4142135623730951
>>> math.pi
3.1415926535897931
```

Strings

- Enclose in quotation marks
  - text = “Hello World‟
- Escape characters with backslash
- Control characters
  - > „\n‟ -> New line
  - > „\t‟ -> Tabulator
- Mark raw strings with r:
  - path_to_file = r"E:\Europe\cities.dxf"

```
>>> print "Hello World"
Hello World
>>> print 'Hello World'
Hello World
>>> print "Hello \“World\""
Hello "World"
>>> print "Hello \n World"
Hello
World
>>> print "Hello \t World"
Hello          World
>>> print r"Hello \t World"
Hello \t World
```

More on Strings

- Use three quotation marks to go over multiple lines
- Just concatenate strings with “+”
- You can even multiply strings with “*”
- Use the built-in function *`len()`* to get the length of a string
- Strings are indexed, use square brackets to access parts
  - My_string[start_pos:end_pos]

Strings – Examples

```
>>> print '''More than
     one line'''
More than
one line

>>> print 'Part one," + ' Part two'
Part one, Part two

>>> print "FME!" * 3
FME! FME! FME!

>>> name = „Tino"
>>> print len(name)
4
```

```
>>> greeting = "Hello World!"
>>> print greeting[4]
o
>>> print greeting[3:7]
lo W
>>> print greeting[3:-2]
lo Wor
>>> print greeting[:7]
Hello W
>>> print greeting[3.0:5]
Traceback (most recent call last):
  File "<pyshell#47>", line 1, in ?
    print greeting[3.0:5]
TypeError: slice indices must
be integers
```

Lists

- Lists can contain elements of different types
  - `mylist = ["Tino", 1, 3.5]`
- Recursive lists – lists in lists
- CreationDirectly
  - Many functions return lists
  - Empty list
    - myList[]
    - myList = list()

Lists - Continuation

- Sorting, Appending, Inserting

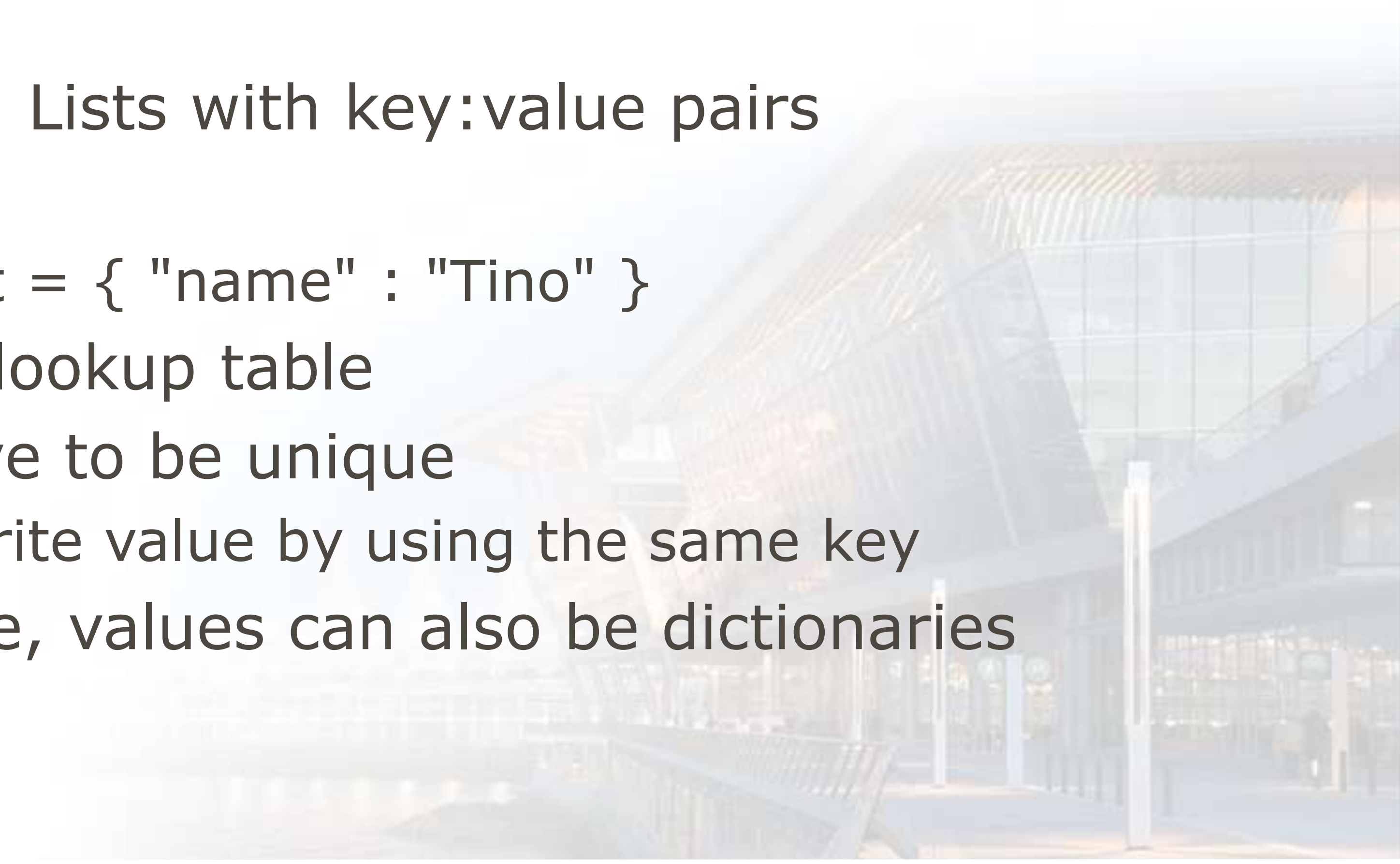
```
>>> myList = ["Stefan", "Hubert", "Katharina", "Maria", "Monika"]
>>> dir(myList)
['append', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
>>> myList.sort()
>>> myList
['Hermann', 'Hubert', 'Katharina', 'Maria', 'Monika', 'Stefan']
>>> myList.reverse()
>>> myList
['Stefan', 'Monika', 'Maria', 'Katharina', 'Hubert']
>>> myList.append("Theresia")
>>> myList.insert(3, 5835)
>>> myList
['Stefan', 'Monika', 'Maria', 5835, 'Katharina', 'Hubert', 'Theresia']
```

Pythonlists – More methods

- Deleting, Overwriting, Counting

```
>>> myList.append("Stefan")
>>> myList
['Stefan', 'Monika', 'Maria', 5835, 'Katharina', 'Hubert', 'Theresia', 'Stefan']
>>> myList.count("Stefan")
2
>>> myList.count("Monika")
1
>>> myList.remove("Stefan")
>>> myList
['Monika', 'Maria', 5835, 'Katharina', 'Hubert', 'Theresia', 'Stefan']
>>> myList[2] = "Matthias"
>>> myList
['Monika', 'Maria', 'Matthias', 'Katharina', 'Hubert', 'Theresia', 'Stefan']
```

Dictionaries



- Unsorted Lists with key:value pairs
- Creation
  - myDict = { "name" : "Tino" }
- Used as lookup table
- Keys have to be unique
  > Overwrite value by using the same key
- Recursive, values can also be dictionaries

Dictionaries - Continuation

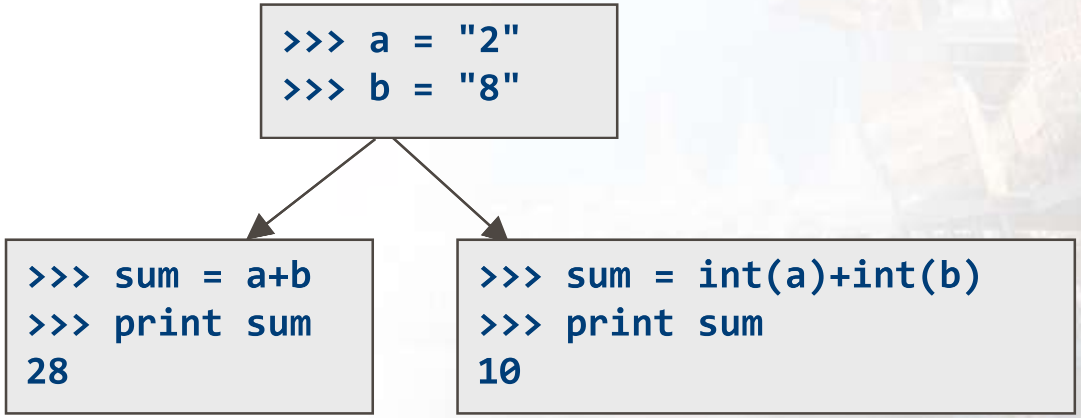
- Access keys and values seperately
- Delete values: del dic[key]

```
>>> myDic = {'Mother':'Maria','Father':'Hermann','Son':'Hubert'}
>>> dir(myDic)
['clear', 'copy', 'get', 'has_key', 'items', 'keys', 'popitem',
'setdefault', 'update', 'values']

>>> myDic.keys()
['Mother', 'Son', 'Father']
>>> myDic.values()
['Maria', 'Hubert', 'Hermann']
>>> myDic.items()
[('Mutter', 'Maria'), ('Sohn', 'Hubert'), ('Vater', 'Hermann')]
```

Casting (Changing the var type)

- Sometimes you want to explicitly change a variable
- Use var type name
  - e.g. int(), float(), str(), list()

```
>>> name = "Marko"
listName = list(name)
>>> listName
['M', 'a', 'r', 'k', 'o']
```



```
>>> a = "2"
>>> b = "8"
```

```
>>> sum = a+b
>>> print sum
28
```

```
>>> sum = int(a)+int(b)
>>> print sum
10
```

Exercise 1.3 – 1.6

- Get to know those different variable types

2014 International User Conference
Vancouver, B.C. Canada

Chapter 2

- Conditions and loops

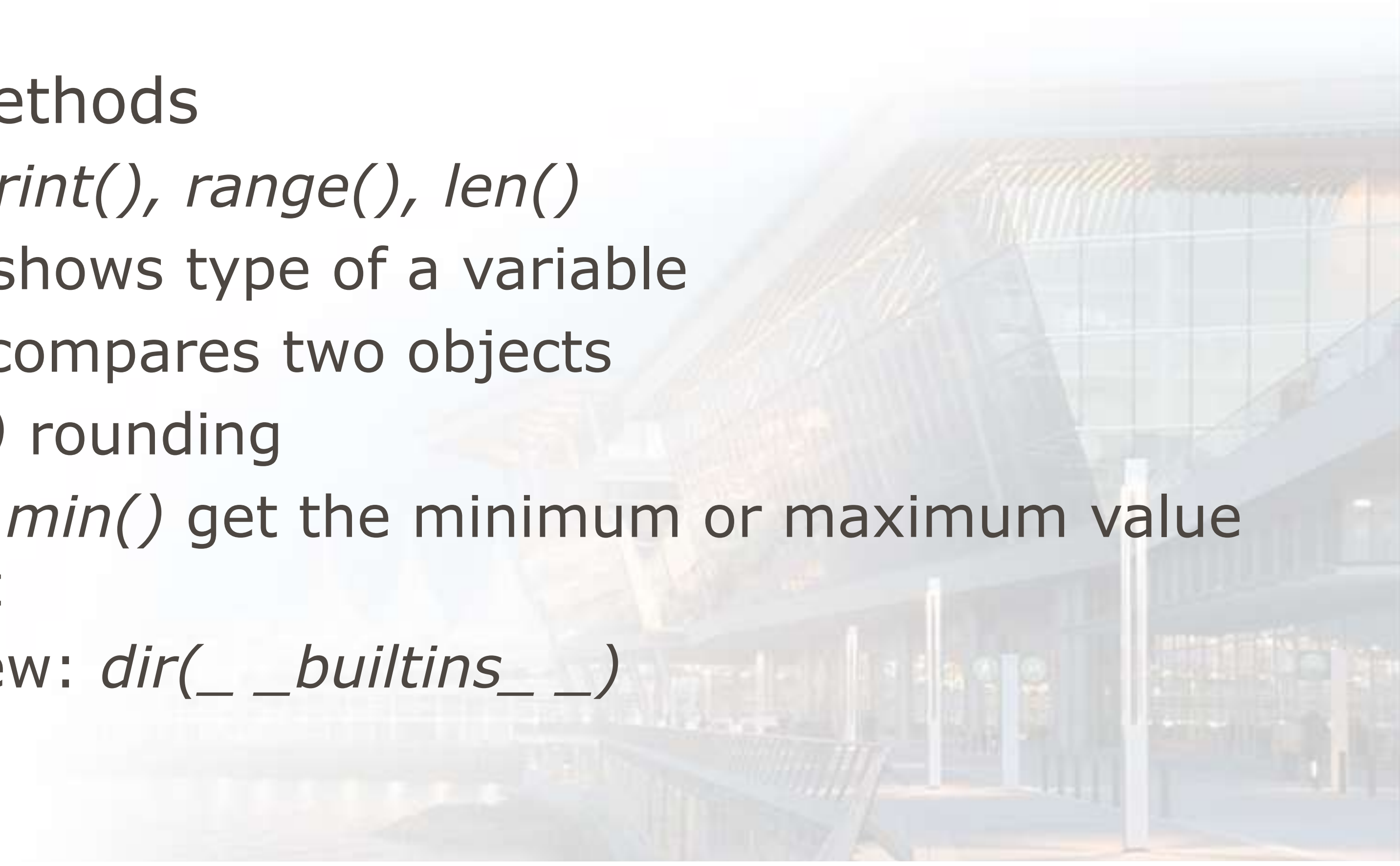
CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content forenoon

- Chapter 0: Introduction to Python
- Chapter 1: Variables, Lists, Dictionaries
- Chapter 2: Loops and conditions
- Chapter 3: FME Workspaces and Python

Builtins



- Builtin methods
  - > *dir(), print(), range(), len()*
  - > *type()* shows type of a variable
  - > *cmp()* compares two objects
  - > *round()* rounding
  - > *max(), min()* get the minimum or maximum value of a list

  →Overview: *dir(_ _builtins_ _)*

Conditions – if/else

- Do different things
- Keywords: *if, elif, else*
  - ":" and Intendation define your code logic
- No code for one condition
  - Use keyword *pass*

```
# Schema
if <Condition>:
    Statement 1
    …
elif  <Condition >:
    Statement 2
else:
    …
<Normal codepath>
```

Comparisons

- Standard: ==, <, >, <=, >=, !=, <>
- Operator „*is*“ compares, if objects are the same
- Operator „*in*“: Does a list contain the element

```
>>> a = range(0,10)
>>> b = range(0,10)
>>> a == b
True
>>> a is b
False
>>> a = b
>>> a is b
True
```

```
>>> a = [0, 1, 2, 3, 4, 5, 6, 7, 8,
9]
>>> 5 in a
True
>>> -5 in a
False
>>> c = 3
>>> c <> 4 # c !=4
True
```

Boolean operators

- *„and"* => logical and
- *„or"* => logical or
- *„not"* => negation

```
5 > 6
False
5 > 6 or 10 > 2
True
5 > 6 and 10 > 2
False
```

```
>>> user = "Maria"
>>> if not user == „Marko":
       print "You are not Marko."

You are not Marko."
```

For loops

- Repeat some code multipletime
- Iterate over lists
- Again, use ":" and intendation for structure

```
>>> sum = 0
>>> a = range(1,5)
>>> for j in a:
        sum = sum + j
        print sum
1
3
6
10
```

While loop

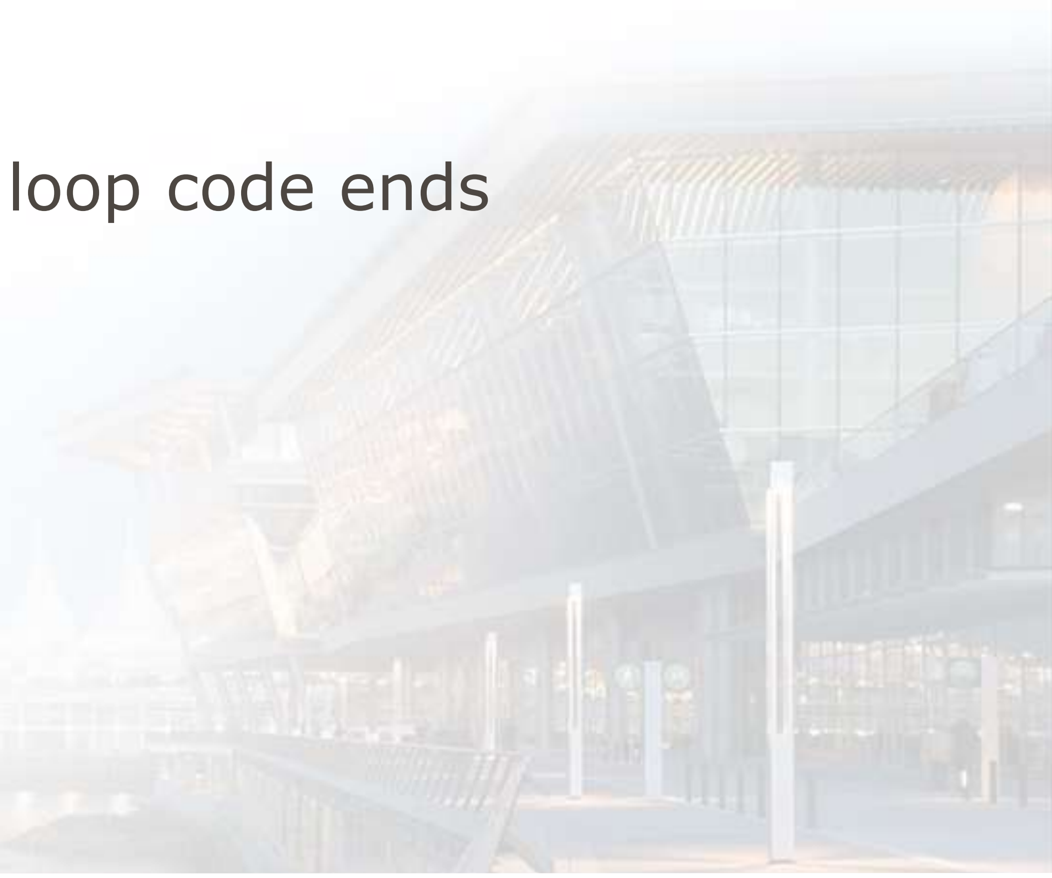
- Does also repeat some code part
- Repeats as long as the condition is True

```
>>> transformer = ["Creator", "Tester", "Inspector",
"Sampler"]
>>> i = 0
>>> while i < len(transformer):
        print i+1,". transformer:", transformer[i]
        i = i + 1
1 . transformer : Creator
2 . transformer : Tester
3 . transformer : Inspector
4 . transformer : Sampler
```

Exit Loop early

- Keyword: break
- Continues where the loop code ends

```
a = range(0,100)  #0..99
for i in a:
    if i == 50:       #gefunden
        print "Found 50!"
        break
print "Normal codepath."
```



Go to next loop iteration

- Keyword: continue
- Skips code, continues with next iteration

```
#Only even numbers
a = range (1, 11)
i = 0
for i in a:
    if i%2 != 0:      #Modulo
        continue
    print i
```

```
2
4
6
8
10
```

Help

- Local Python Help
  - > Start > Programs > Python 2.7 > Python Manuals
- Internet
  - > http://python.org/
    Documentation, PythonWiki ….

Exercise 2

- Write a guess-the-number-game

2014 International User Conference
Vancouver, B.C.
Canada

Chapter 3

- FME Workbench and Python

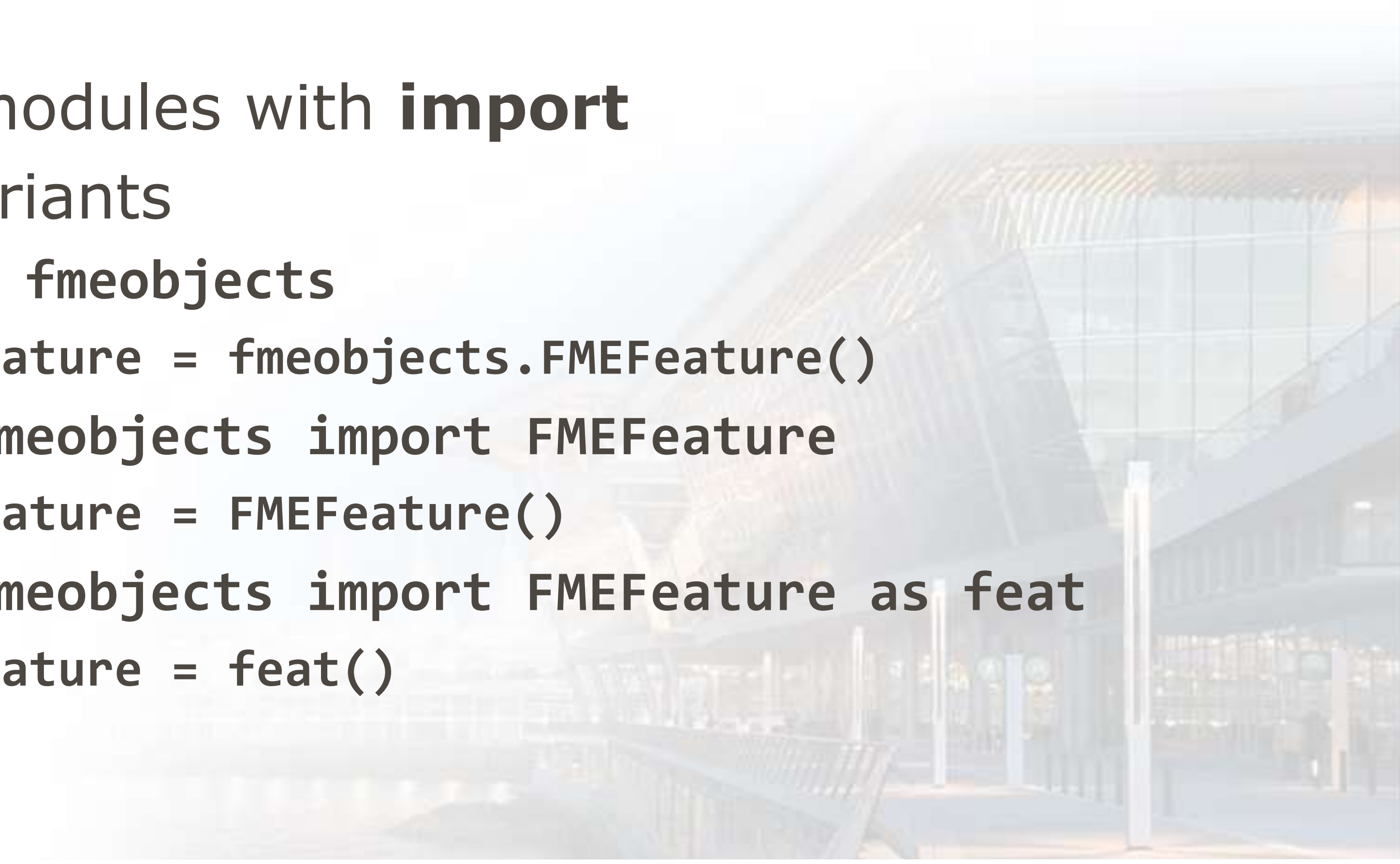
CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content forenoon

- Chapter 0: Introduction to Python
- Chapter 1: Variables, Lists, Dictionaries
- Chapter 2: Loops and conditions
- Chapter 3: FME Workspaces and Python

Modules



- Import modules with **import**
- Some variants
  - `import fmeobjects`
    - `myFeature = fmeobjects.FMEFeature()`
  - `from fmeobjects import FMEFeature`
    - `myFeature = FMEFeature()`
  - `from fmeobjects import FMEFeature as feat`
    - `myFeature = feat()`

Python functions

- Keyword: **def**
- Need a name
- Parameters in round brackets

```
def processFeature(feature):
    feature.setAttribute("name","Tino")
```

Python Classes

```
class FeatureProcessor(object):
    def __init__(self):
        self.constant = "Tino"
    def input(self,feature):
        feature.setAttribute("Name",self.constant)
        self.pyoutput(feature)
    def close(self):
        pass
```

FMEObjects Documentation

- <FMEHOME>/fmeobjects/python/apidoc/index.html
- Have a look especially at
  - FMELogFile
  - FMEFeature

FMELogFile()

- Much better than doing print("My message")
- Gets output to the Workbench Log window and the default logfile
- Use different severity levels:
  - fmeobjects.FME_INFORM
  - fmeobjects.FME_WARNING
  - …
- Hint: Activate "Log timestamp information" in the FME Runtime options

Exercise 3

- Add a Startup script
- Use a function and a class implementation in the PythonCaller

2014 INTERNATIONAL USER CONFERENCE
VANCOUVER, B.C. CANADA

Chapter 4

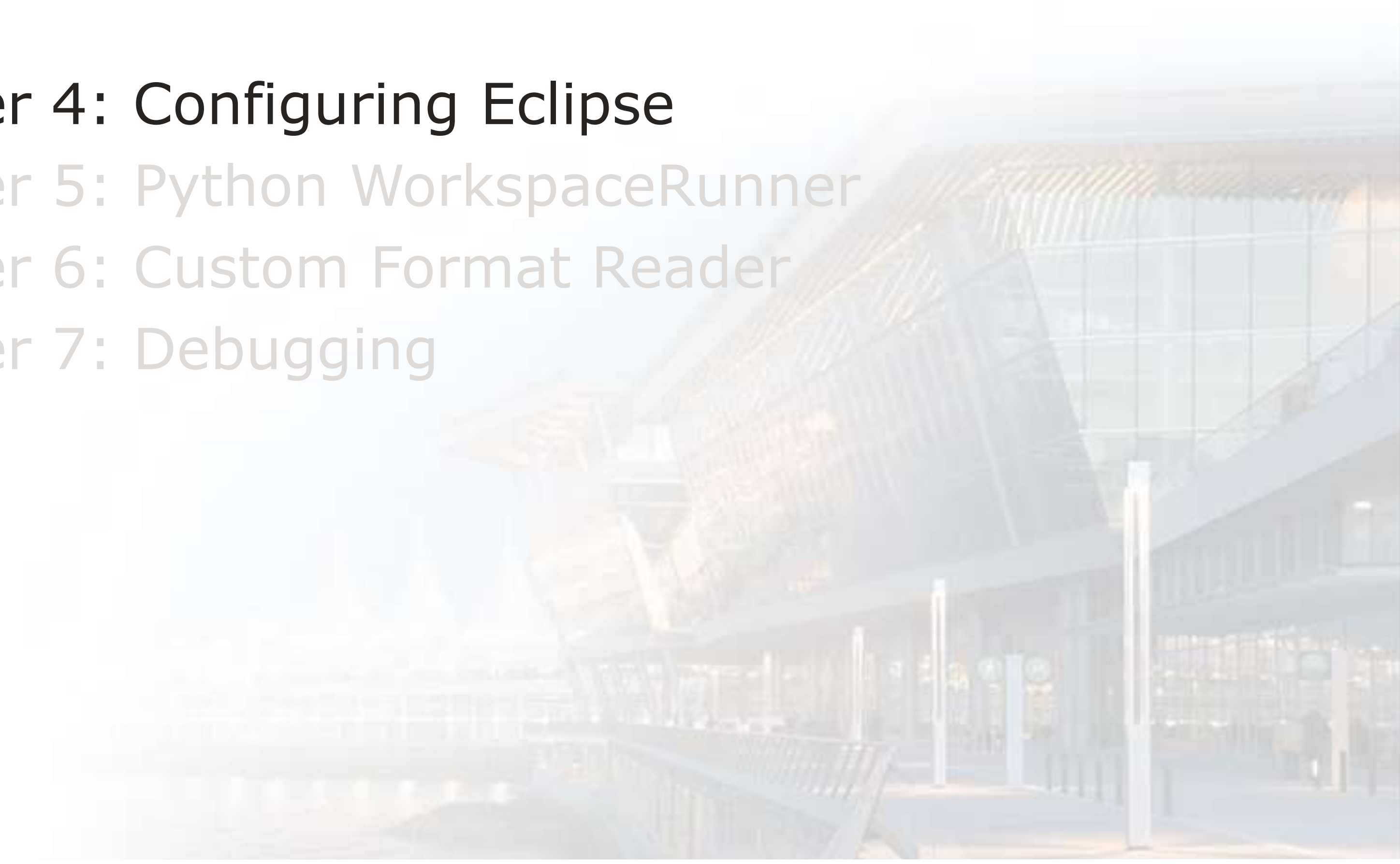
- Configuring Eclipse + PyDev

CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content afternoon

- Chapter 4: Configuring Eclipse
- Chapter 5: Python WorkspaceRunner
- Chapter 6: Custom Format Reader
- Chapter 7: Debugging

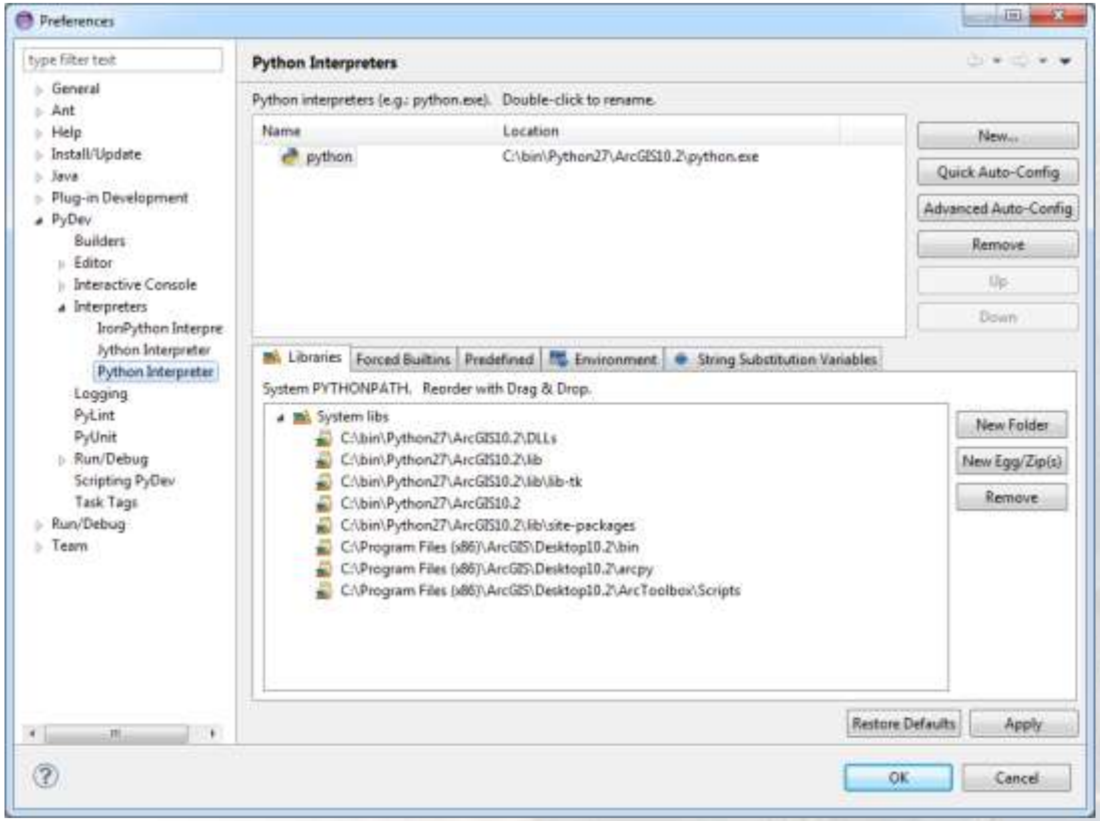


Eclipse

- Very extensive programming IDE
- Python Plugin PyDev
- CVS/SVN/GIT Support

- On startup choose **C:\PythonTraining\eclipse_workspace** as workspace directory

Configuring Python Interpreter


Python Interpreters

Python interpreters (e.g.: python.exe). Double-click to rename.

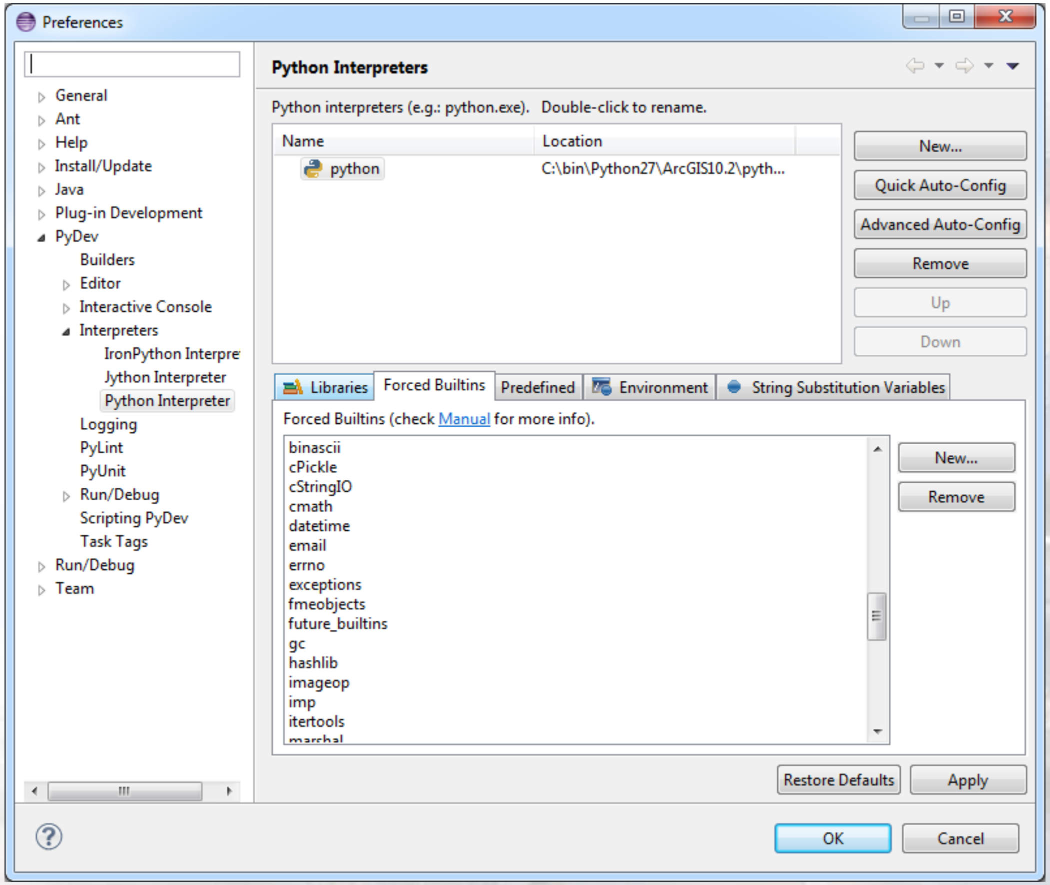
Name	Location
python	C:\bin\Python27\ArcGIS10.2\python.exe

System PYTHONPATH. Reorder with Drag & Drop.

```
System libs
    C:\bin\Python27\ArcGIS10.2\DLLs
    C:\bin\Python27\ArcGIS10.2\lib
    C:\bin\Python27\ArcGIS10.2\lib\lib-tk
    C:\bin\Python27\ArcGIS10.2
    C:\bin\Python27\ArcGIS10.2\lib\site-packages
    C:\Program Files (x86)\ArcGIS\Desktop10.2\bin
    C:\Program Files (x86)\ArcGIS\Desktop10.2\arcpy
    C:\Program Files (x86)\ArcGIS\Desktop10.2\ArcToolbox\Scripts
```

Configure FMEObjects

- To get code assist, the FMEobject module has to be configured as Forced Builtin



Python Interpreters

Python interpreters (e.g.: python.exe). Double-click to rename.

Name	Location
python	C:\bin\Python27\ArcGIS10.2\pyth...

Forced Builtins (check Manual for more info).

```
binascii
cPickle
cStringIO
cmath
datetime
email
errno
exceptions
fmeobjects
future_builtins
gc
hashlib
imageop
imp
itertools
```

Exercise 4

- Configure Eclipse
- Run the TestFME Project

2014 International User Conference
Vancouver, B.C. Canada

Chapter 5

- FMEWorkspaceRunner

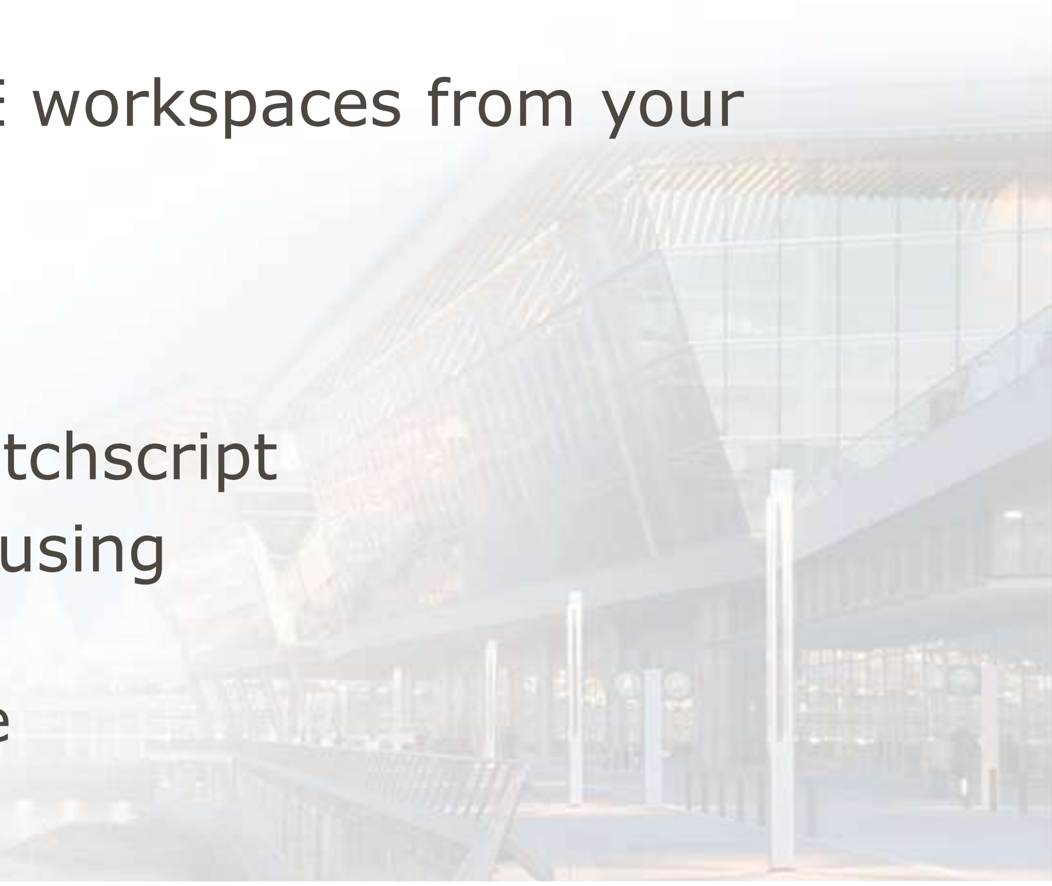
CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content afternoon

- Chapter 4: Configuring Eclipse
- Chapter 5: Python WorkspaceRunner
- Chapter 6: Custom Format Reader
- Chapter 7: Debugging

FMEWorkspaceRunner



- Helps you to run FME workspaces from your Python code
- Automation
- Batch Processing
- More flexible than Batchscript
- Much less code than using
  - **os** module
  - **subprocess** module

FMEWorkspaceRunner

- Use the Python WorkspaceRunner interface to retrieve information from workspaces:
  - getPublishedParamNames(workspace)
  - getParamValues(workspace, paramName)
  - getParamDefaultValue(workspace, paramName)
  - getParamLabel (workspace, paramName)

FMEWorkspaceRunner

- Use the Python WorkspaceRunner interface to run your FME workspaces:
  - `run(workspace)`
  - `withParameters(workspace, parameters)`
  - `promptRun(workspace)`

Exercise 5

- Open the DemoWorkspaceRunner project in Eclipse
- Your goal is to complete the python code to get a successful run of the workspace

2014 International User Conference
Vancouver, B.C. Canada

Chapter 6

- Custom Format Reading

CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content afternoon

- Chapter 4: Configuring Eclipse
- Chapter 5: Python WorkspaceRunner
- Chapter 6: Custom Format Reader
- Chapter 7: Debugging

Custom Format Schema

FEATURE	Begin of a new feature
ID:638775314	Attribute:Value
AMENITY:school	Attribute:Value
NAME:Lord Byng High School	Attribute:Value
-123.1929566	Geometry x
49.2588828	Geometry y

PythonCaller

- Each text line enters as one feature
- For each line you have to decide (if…):
  - It is a "new Feature" line
  - Contains an attributename:attributevalue pair
  - Contains the x or y coordinate

- Write code in CustomReader.py
- Use this file with PythonCaller (Entrypoint)

Creating a FMEGeometry

- Create a Feature:

```
myFeature = fmeobjects.FMEFeature()
```

- Create a Geometry:

```
geom = fmeobjects.FMEPoint(x, y)
```

- Attach Geometry to Feature:

```
myFeature.setGeometry(geom)
```

Exercise 6

- Open the CustomReader project in Eclipse
- Open the contained workspace with FME

- Please replace the code in CustomReader.py with

http://goo.gl/5hb3gx

2014 International User Conference
Vancouver, B.C. Canada

Chapter 7

- Debugging

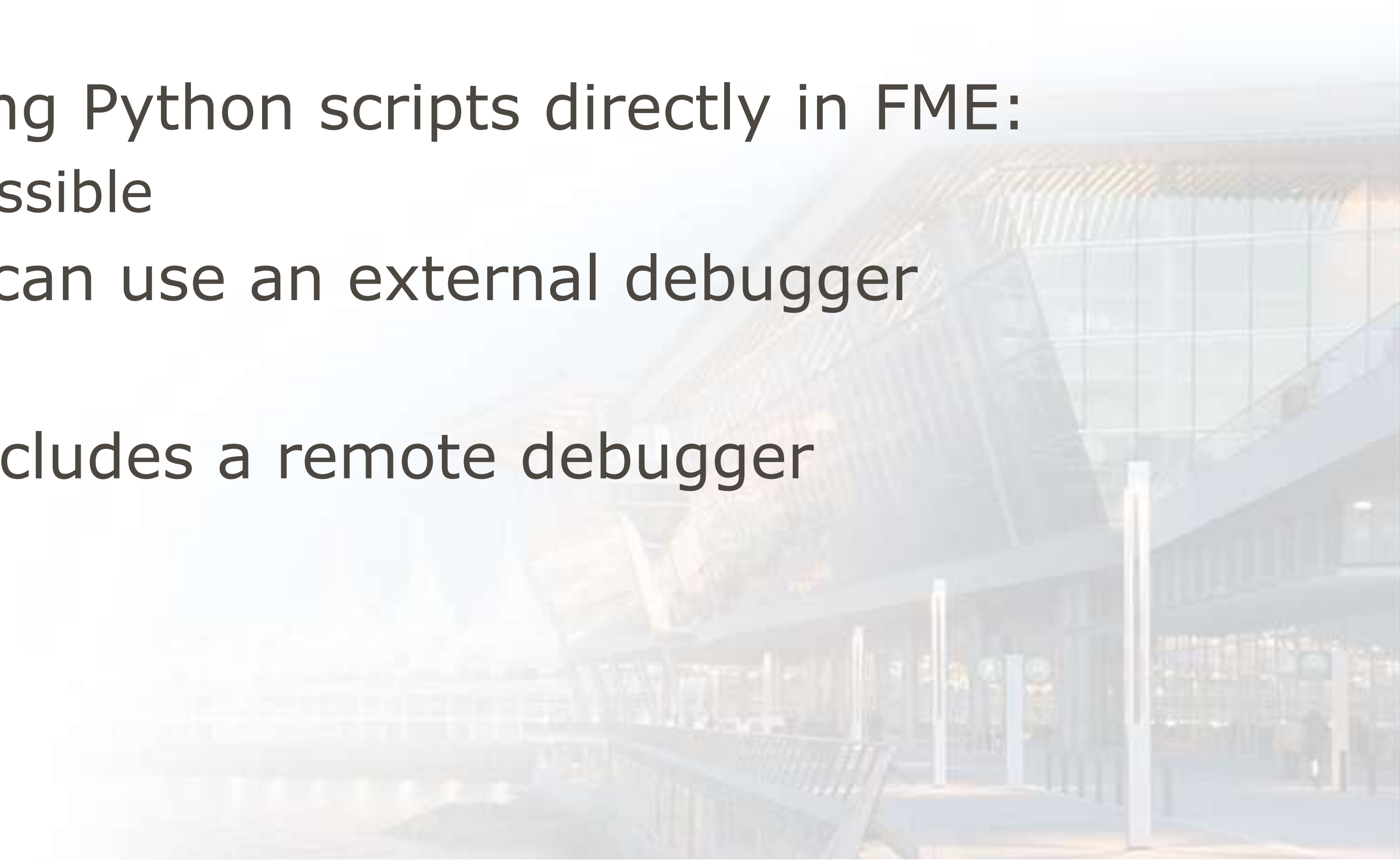
CONNECT. TRANSFORM. AUTOMATE.

SAFE SOFTWARE

Course content afternoon

- Chapter 4: Configuring Eclipse
- Chapter 5: Python WorkspaceRunner
- Chapter 6: Custom Format Reader
- Chapter 7: Debugging

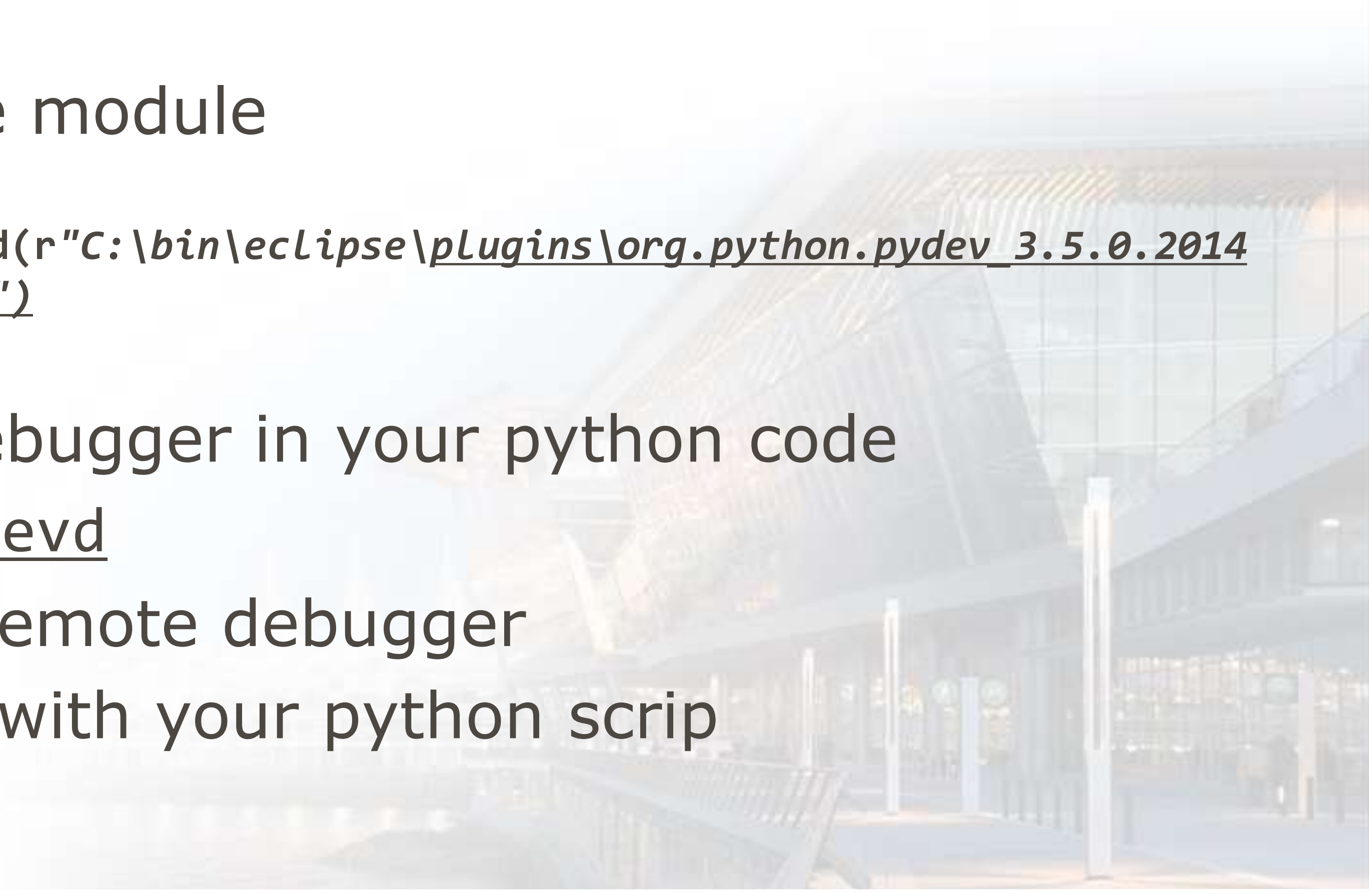
Debugging



- Debugging Python scripts directly in FME:
  - Not possible
- But you can use an external debugger

- PyDev includes a remote debugger

Steps to debug



1. Import the module

```
import sys
sys.path.append(r"C:\bin\eclipse\plugins\org.python.pydev_3.5.0.2014
05201709\pysrc")
import pydevd
```

2. Call the debugger in your python code

```
import pydevd
```

3. Start the remote debugger
4. Start FME with your python scrip

Thank You!

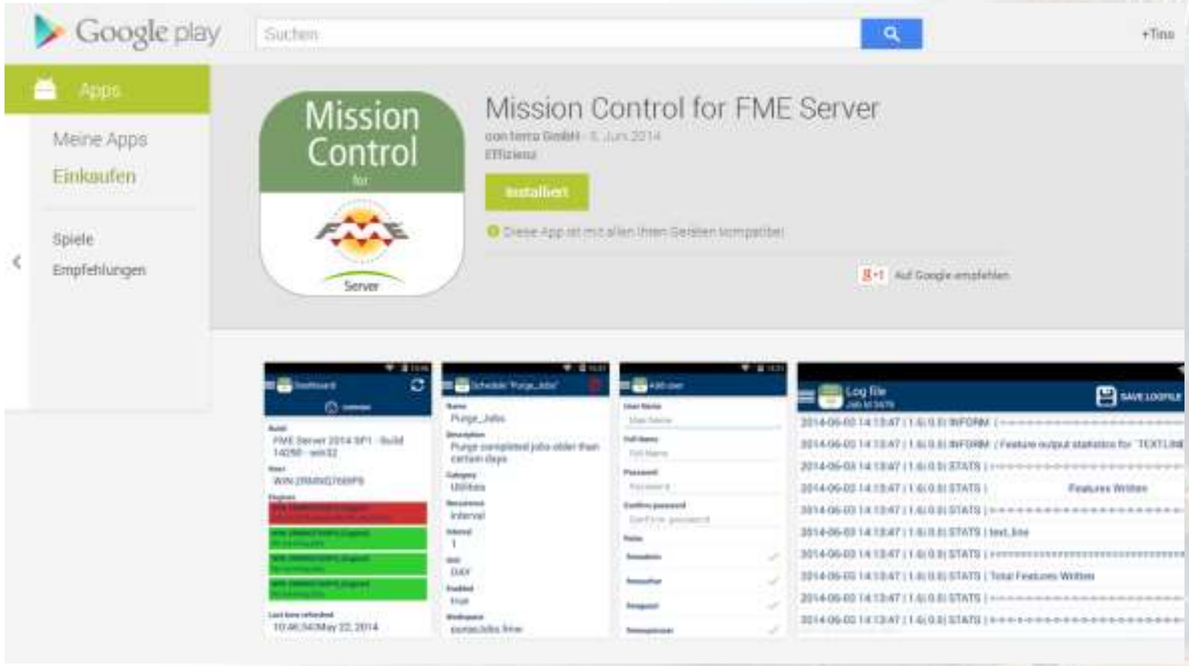
- Questions?

- For more information:
  - Tino Miegel, t.miegel@conterra.de
    @Tmiegel

- **For a certificate of participation mail train@safe.com with the course name**
- **Have a safe trip home!**

Mission Control 4 FME Server

- http://mc4fme.com => G+ Beta test group



Mission Control for FME Server

con terra GmbH - 5. Juni 2014

Effizienz

Diese App ist mit allen Ihren Geräten kompatibel

```
2014-06-03 14:13:47 | 1.6| 0.0| INFORM | -----------------------------
2014-06-03 14:13:47 | 1.6| 0.0| INFORM | Feature output statistics for `TEXTLINE
2014-06-03 14:13:47 | 1.6| 0.0| STATS | -----------------------------
2014-06-03 14:13:47 | 1.6| 0.0| STATS |                    Features Written
2014-06-03 14:13:47 | 1.6| 0.0| STATS | =============================
2014-06-03 14:13:47 | 1.6| 0.0| STATS | text_line
2014-06-03 14:13:47 | 1.6| 0.0| STATS | =============================
2014-06-03 14:13:47 | 1.6| 0.0| STATS | Total Features Written
2014-06-03 14:13:47 | 1.6| 0.0| STATS | -----------------------------
2014-06-03 14:13:47 | 1.6| 0.0| STATS | =============================
```