Using Python in SpatialNET

Ben Morgan

Software Developer

spatialinfo
CLEARING THE NETWORK CLOUD

What is Python?

- A programming language
- Doesn’t require compiling
- Easy to read and write
- A concise language
- SpatialNET embeds standard C Python 2.7
- Download from www.python.org

Python Example

```
temperatures = [86, 75, 32, 104]
for f in temperatures:
    c = (f-32) * 5.0 / 9.0
    print “Farenheit”, f
    print “Celsius”, round(c, 1)
```

What Can You Do With Python?

- **Read/write entities from database**
- **Generate Excel/HTML reports**
- **Data integrity testing**
- **Hooks into existing commands**
  - Pre-populate fields on add windows
  - Validate fields on add/edit windows
  - React to entities being edited
- **And lots more…**

spatial info
CLEARING THE NETWORK CLOUD

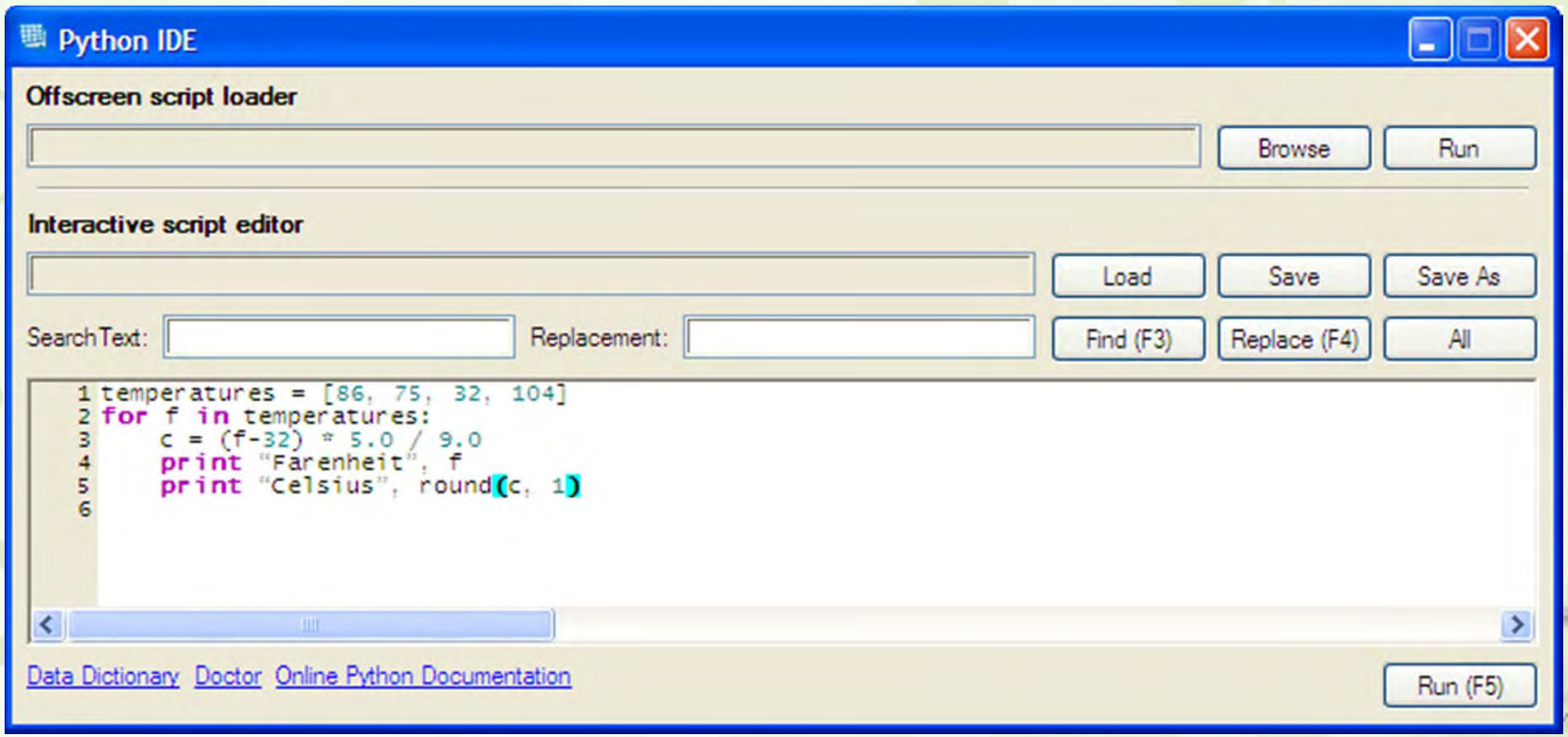
How We Use Python In spatialNET

- **Doctor tests**
  - Database schema test
- **Trace, splice and tray reports**
- **Various other reports**
- **Google Earth export**
- **Distance Tool**
- **Dark fiber trace**
- **Etc.**

spatial info
CLEARING THE NETWORK CLOUD

Where You Can Use It

- Adhoc scripts in Python IDE
- As an administrator, open up Debug Panel
- Open Scripting > Python



Where You Can Use It

- **ACAD button/menu definitions**
  - (snet-python “path/to/file.py”)
- **Run Python file in spatialNET from command line**
  - spatialNET /python flag
  - Suitable for batch processes

Where You Can Use It

- **Custom autoload directory**
  - <spatialNET>\python\spatialnet\custom\autoload
  - Automatically loaded on startup
  - Suitable for registering hooks into spatialNET
- **Automatically scanned report directories**
  - Generic reports
  - Trace, splice, tray reports
  - Doctor reports

Example

- **Update state (i.e. address details, not status) for splice cases inside boundary**
  - When user selects a boundary and runs script
  - Whenever a boundary is changed
- **We could also update town/city**

Access the Current Selection

- gdm.selected_entity()
- Access the current selection
- Can specify a type to check
- E.g. to check a single boundary is selected:

```
import gdm
boundary = gdm.selected_entity(
    Check="FM_BOUNDARY")
```

Access Entities

- eam.scan – find entities matching conditions
- E.g. to fetch all splice cases

```
import eam
print eam.scan("SPLICE_CASE")
```

- Geometric conditions – e.g. splice cases inside a boundary
- Attribute-based conditions – e.g. type of splice case is Unknown

Find Splices

- Find splice cases inside boundary

```
splices = eam.scan("SPLICE_CASE",
    [],
    ["NODAL_LOCATION",
     eam.SCAN_ENCLOSED_BY,
     boundary])
```

Access Entities

```
print “Boundary state”
print boundary.fdm_polygon_state
Print “Splice states”
for splice in splices:
    print splice.fdm_state
```

Updating Entities

- Updates must happen inside spatialNET actions and transactions
- Updates are performed as part of the current job and are undoable in the current job
- Use gdm.action and gdm.transaction

Updating an Entity

```
splice = gdm.selected_entity()
c = "SPLICE_CASE"
b = "edit"
with gdm.action(c, b):
  with gdm.transaction(c, b):
     splice.fdm_state = "CO"
```

Updating Entities

```
c = "SPLICE_CASE"
b = "edit"
with gdm.action(c, b):
   for splice in splices:
      with gdm.transaction(c, b):
           splice.fdm_state = \
                boundary.fdm_state
```

Putting It Together

```
import eam, gdm
boundary = gdm.selected_entity(
      Check="FM_BOUNDARY")
splices = eam.scan("SPLICE_CASE",[],
      ["NODAL_LOCATION", eam.SCAN_ENCLOSED_BY,
       boundary])
c = "SPLICE_CASE"
b = "edit"
with gdm.action(c, b):
   for splice in splices:
      with gdm.transaction(c, b):
            splice.fdm_state = \
                  boundary.fdm_state
```

Steps Hooks

- Perform an action when something is happening in spatialNET (e.g. add or edit of a piece of equipment)
- Log information
- Update other entities based on this one
- Validate user input

spatial info
CLEARING THE NETWORK CLOUD

Using an Edit Hook

```
import gdm, eam
def hook(classname, behavior, settings, stage):
  if stage != 0: return
  boundary = gdm.selected_entity()
  splice_cases = eam.scan("SPLICE_CASE", [],
     ["NODAL_LOCATION", eam.SCAN_ENCLOSED_BY,
          boundary])
  for splice_case in splice_cases:
     splice_case.fdm_state = \
          boundary.fdm_polygon_state

gdm.register_steps_hook("FM_BOUNDARY", "edit",
  "UPDATE_AFFECTED", hook)
```

Ensure Cable Name Entered

```
import SPATIALnet, eam, gdm
def check_name(classname, behavior,
  settings, stage):
  name = eam.editbuffers.fdm_cable_name
  # stop if it isn't a valid cable name
  if not name:
     gui.ShowErrorMessage(
          "Please enter a cable name")
     SPATIALnet.user_cancelled()
```

Ensure Cable Name Entered

```
gdm.register_steps_hook(
  "FIBER_CABLE_UNCON", "add",
  "PRE_STEPS", check_name)
gdm.register_steps_hook(
  "FIBER_CABLE_UNCON", "edit",
  "PRE_STEPS", check_name)
```

Reporting

- **reports package**
- **reports.excel module**
  - Supports reading and writing excel
- **reports.html module**
- **reports.generic module**
  - Writes to HTML and Excel
  - Easy to use reporting
  - Mostly used for writing tables

Generic Reports

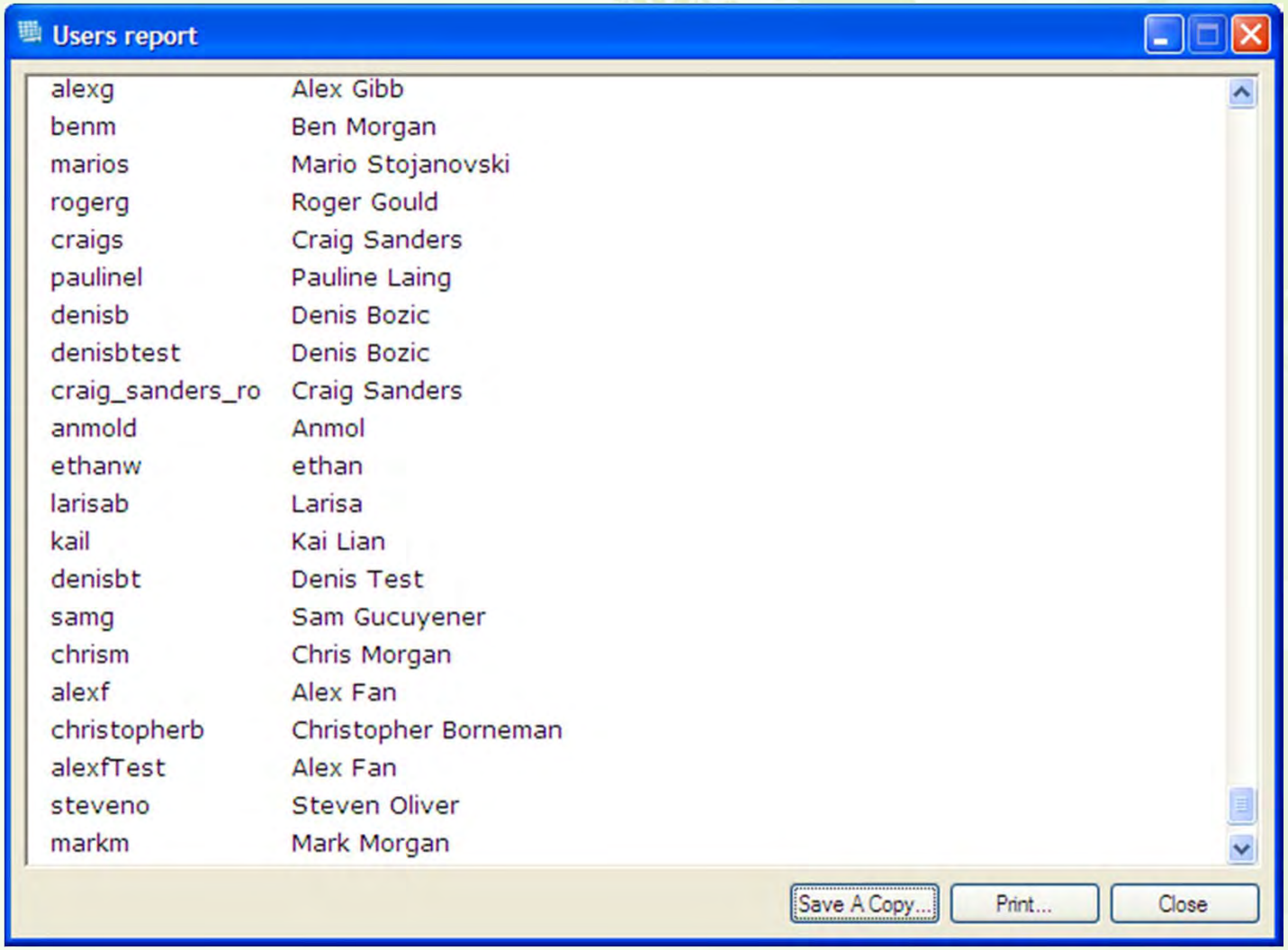
```
from reports.generic import *
import eam

report = Report("Users report")
sheet = report.addSheet("Users report")
table = sheet.addTable()
row = table.addRow()
row.addHeader("User")
row.addHeader("Real name")
for user in eam.scan("scm_user"):
      row = table.addRow()
      row.addCell(user.eam_username)
      row.addCell(user.real_name)

writer = GenericWriter.getWriterForFile(report,
            r"c:\file.html")
writer.save_and_show()
```

spatial info
CLEARING THE NETWORK CLOUD

Generic Reports



ACAD

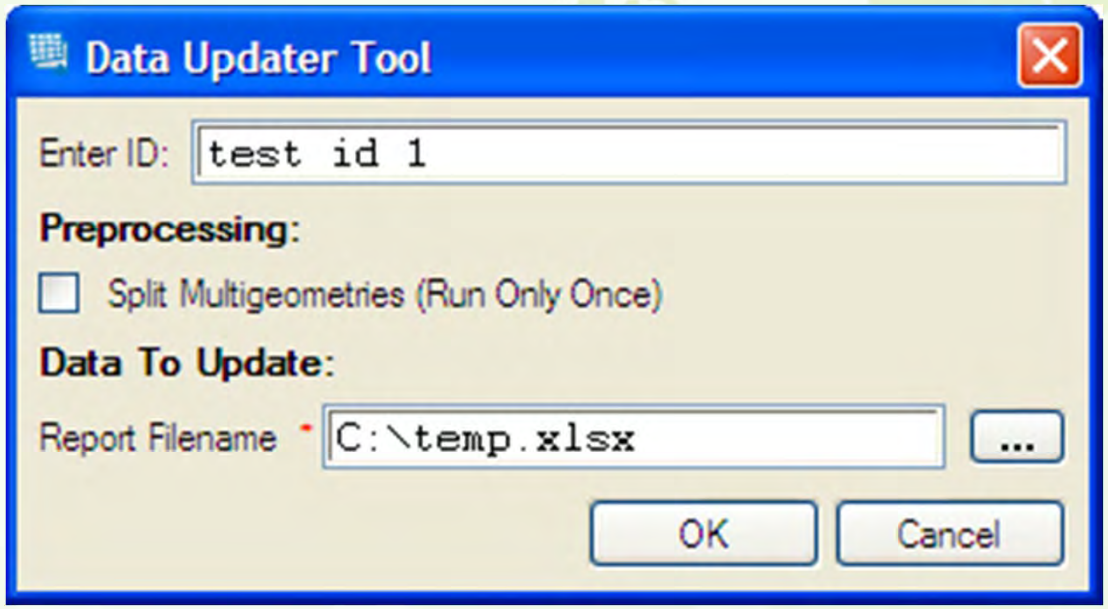
- **ACAD module**
- **Walk through AutoCAD drawing**
- **Access properties of**
  - Drawing
  - Objects in drawing (e.g. read block attributes)
- **Populate AutoCAD drawing programmatically**
- **Interact with AutoCAD through COM**

GUI

- gui module
- Provides graphical user interface support
- Ask user for options
- Simple and easy to configure
- Use text boxes, checkboxes, pulldowns and ask the user for filename

GUI

- **14 lines of code**



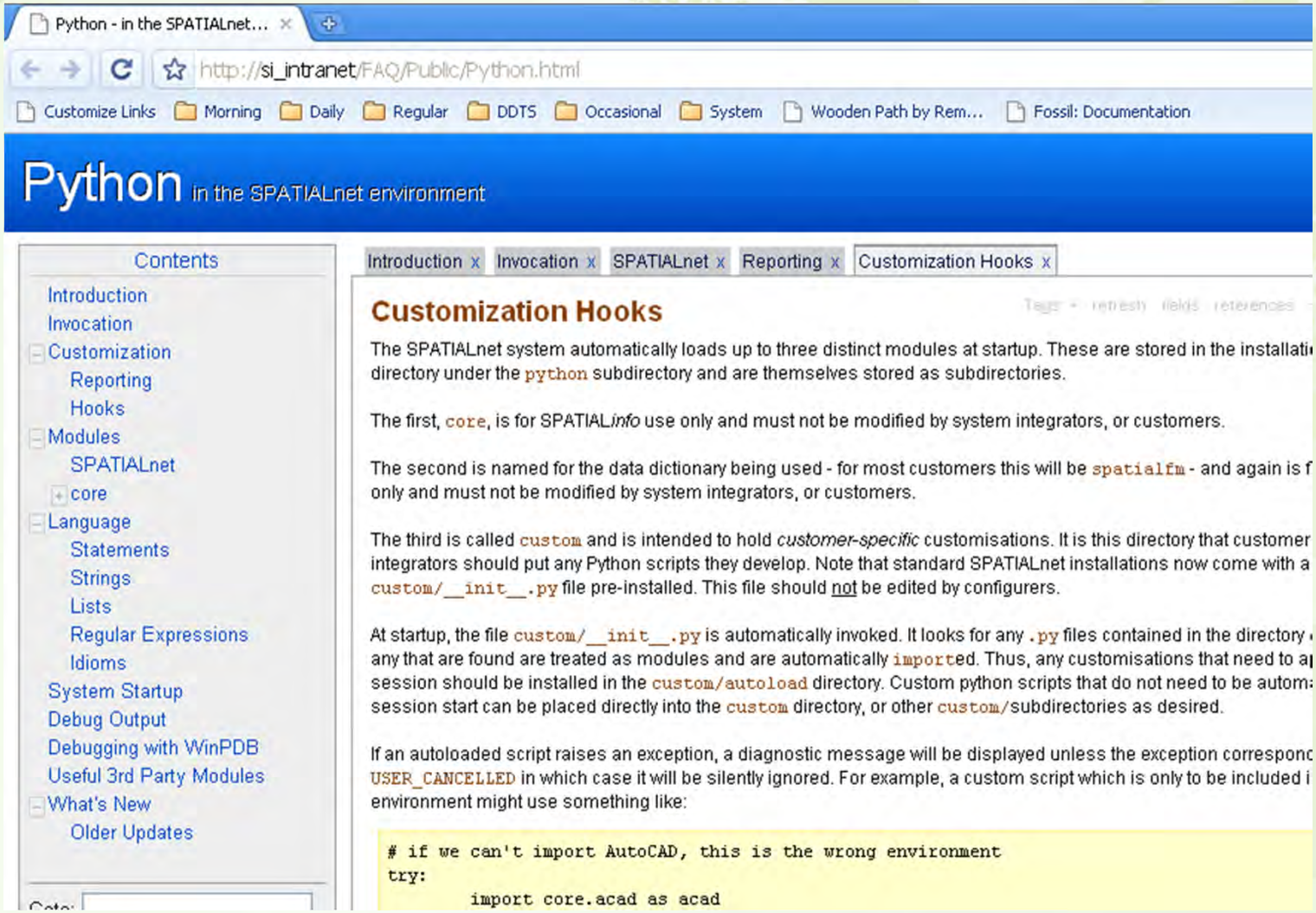
JMS

- JMS module
- JMS = Job Management System
- Find what has been added/changed/deleted in job

Caveats

- **Limited in some areas**
- **Python support is spatialNET only – not spatialWEB or spatialOFFLINE**
- **Some changes between versions of spatialNET**
  - Especially if you port back to earlier version
- **spatialNET has a very complex data model**
  - Can be difficult to work with
- **Large parts of spatialNET undocumented**

Documentation



spatial info
CLEARING THE NETWORK CLOUD

Conclusion

- **Customize spatialNET with Python**
- **Control/make your own customizations**
- **Supports a wide variety of customization tasks**
- **Let us know if you are using it**
  - Ensure backwards compatibility

spatial info
CLEARING THE NETWORK CLOUD

A SPATIALinfo Community Portal?

- **Proposal for an online portal to facilitate collaboration on SPATIALinfo product extensions**
  - Python scripting applications and potentially other product extensions and data in the future
  - Applications can be posted and shared on the portal
  - An interactive blog for users to discuss application development and use
  - Applications will not be supported by SPATIALinfo
  - Employees will participate in the portal and moderate blog discussions
- **Let us know what you think!**

spatial info
CLEARING THE NETWORK CLOUD

THANK YOU

QUESTIONS OR COMMENTS?

spatial info
CLEARING THE NETWORK CLOUD