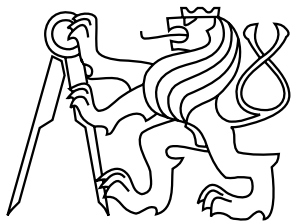
Programování grafických aplikací (BI-PGA), Přednáška č. 7

3D grafika - Blender: Úvod, metodika, využití jazyka Python

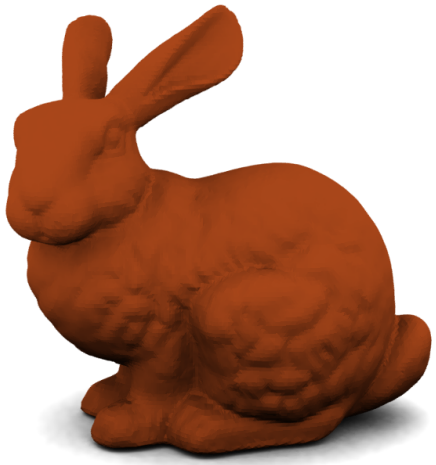
Jiří Chludil

Fakulta informačních technologií
České vysoké učení technické v Praze
https://courses.fit.cvut.cz/BI-PGA/



ZS 2020/2021

Ikony 3D



Historie rozšiřování aplikací pomocí pluginů

3D Studio

- 1990 - Verze 1 - 3D Studio pod OS MS-DOS
- 1996 - Verze 1 - 3D Studio Max pod OS windows
- 2005 - Akvizice Autodeskem
- 2007-16 - Verze 2008-2017
- 2006 - podpora pluginů

Historie rozšiřování aplikací pomocí pluginů

Blender

- 1994 - Verze 1.0 - začal vývoj ve studiu NeoGeo
- 1999 - Verze 1.6x - release pro Windows, PPC
- 2000 - Verze 2.1 - podpora Pythonu
- 2001 - Verze 2.2x - release pro MacOs
- 2002 - Verze 2.25 - vydán jako Open Source
- 2008 - Verze 2.48 - Python skripty
- 2009-11 - Verze 2.5x - Python API
- 2012 - Verze 2.5x - Python API (upgrade)
- 2013 - Verze 2.5x - Python API (nodes. mesh tools)
- 2016 - Verze 2.80 - Změny na API (bez zpětné kompatibility)
- 2020 - Verze 2.90 - Aktuální verze

Co je třeba řešit u zásuvných modulů

- **Typ modulu**
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Typy modulů u Blenderu I

- Vytváření geometrie a objektů
  - Geometrie (Mesh)
  - Křivky
  - Objekty
  - Generování složitějších struktur
- Manipulace s objekty
  - Modifikátory
  - Generování
  - Deformace
  - Simulace
- Import/Export
  - Formáty
  - API

Typy modulů u Blenderu II

- Materiály a textury
  - Materiály
  - Mapování
  - Textury
- Systém
  - 3D zobrazení
  - Text editor
  - Témata
- Rendering
  - Renderer
  - Node
- Animace
- Game engine

Co je třeba řešit u zásuvných modulů

- Typ modulu
- **Nastavení prostředků (konfigurace)**
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Nastavení prostředků (konfigurace)

```
bl_info = {
    "name": "House",
    "description": "Klasický domeček.",
    "author": "Jiří Chludil",
    "version": (1, 0, 0),
    "blender": (2, 77, 0),
    "location": "View3D > Add > Mesh > House",
    "warning": "",
    "wiki_url": "http://adresa .addonu",
    "tracker_url": "http://developer.blender.org...,
    "category": "Add Mesh"
}
```

Co je třeba řešit u zásuvných modulů

- Typ modulu
- Nastavení prostředků (konfigurace)
- **Registrace**
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Registrace I

```
def menu_func(self, context):
    self.layout.operator(MyHouse.bl_idname)
# store keymaps addon_keymaps = []

def register():
    bpy.utils.register_module(__name__)
    bpy.types.INFO_MT_mesh_add.append(menu_func)

def unregister():
    bpy.utils.unregister_module(__name__)
    bpy.types.INFO_MT_mesh_add.remove(menu_func)

if __name__ == "__main__":
    register()
```

Registrace II

v2.79

```
def register():
    bpy.utils.register_module(__name__)

def unregister():
    bpy.utils.unregister_module(__name__)
```

v2.80

```
classes = (AClass, BClass, CClass )

def register():
    from bpy.utils import register_class
    for cls in classes:
        register_class(cls)

def unregister():
    from bpy.utils import unregister_class
    for cls in reversed(classes):
        unregister_class(cls)
```

)

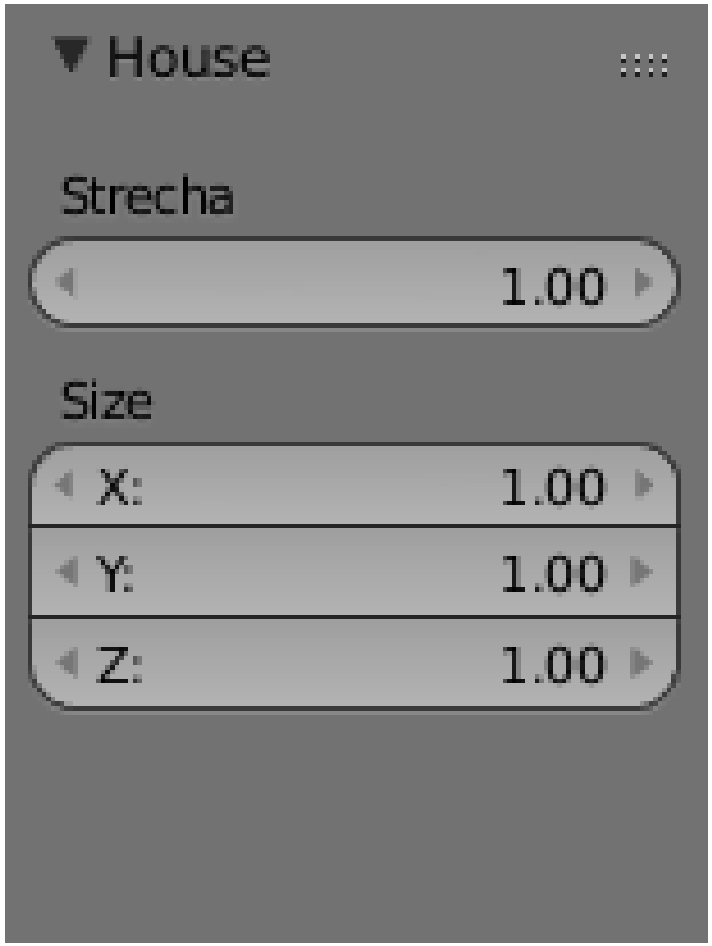
Co je třeba řešit u zásuvných modulů

- Typ modulu
- Nastavení prostředků (konfigurace)
- Registrace
- **Uživatelské rozhraní modulu**
- Použité matematické a grafické funkce
- API rozšiřované aplikace

Uživatelské rozhraní modulu

- Základní typy
  - BoolProperty
  - FloatProperty
  - IntProperty
  - StringProperty
  - EnumProperty
- Vektory
  - BoolVectorProperty
  - FloatVectorProperty
  - IntVectorPropertyty
- Kolekce
  - CollectionProperty
- Strukturální nástroje
  - PointerProperty
  - RemoveProperty

Uživatelské rozhraní modulu - Minimální GUI I



Uživatelské rozhraní modulu - Minimální GUI II

```
roof = bpy.props.FloatProperty(
    name = 'Strecha',
    description = 'vyska strechy',
    default = 1, min = 0.0, max=2.0, step=0.1
    )

size = bpy.props.FloatVectorProperty(
    name = 'Size',
    description = 'Size',
    subtype = 'XYZ',
    default=(1.0, 1.0, 1.0),
    )
```

Uživatelské rozhraní modulu - Pokročilé GUI II

UILayout

- Formátování
  - column
  - column_flow
  - row
  - grid_flow
  - box
  - split
- Elementy gui
  - prop
  - label
  - operator
  - menu
  - separator

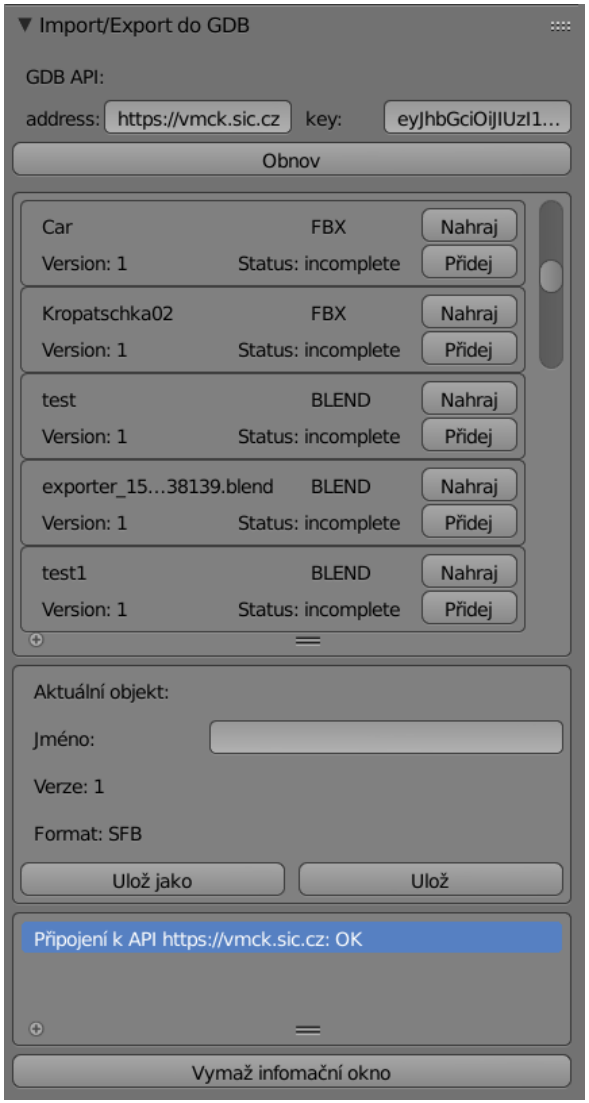
Uživatelské rozhraní modulu - Pokročilé GUI II

```
class ExamplePanel(bpy.types.Panel):
    bl_idname = "myExample"
    bl_label = "Můj Panel"
    bl_space_type = 'VIEW_3D'
    bl_region_type = 'TOOL_PROPS'

    def draw(self, context):
        row = layout.row(align=True)
        row.prop(api, "address")
        row.prop(api, "key")
        row.operator("info.clear")
```

Uživatelské rozhraní modulu -Pokročilé GUI III

```
class ClearInfo(bpy.types.Operator):
    bl_idname = 'info.clear'
    bl_label = 'Název tlačítka'

    def execute(self, context):
        # akce tlačítka
        return {'FINISHED'}
```

Uživatelské rozhraní modulu -Pokročilé GUI IV



Uživatelské rozhraní modulu - Pokročilé GUI IV

```
def draw_item(self, context, layout, data, item, icon,
active_data, active_propname, index):
    if self.layout_type in {'DEFAULT', 'COMPACT'}:
        box = layout.box()
        row= box.split(percentage=0.8)
        col1 = row.column()
        col2 = row.column()
        row1_split = col1.split(percentage=0.7,
align=True)
        row1_col1 = row1_split.column()
        row1_col2 = row1_split.column()
        row2_split = col1.split(percentage=0.5)
        row2_col1 = row2_split.column()
        row2_col2 = row2_split.column()
```

Uživatelské rozhraní modulu - Pokročilé GUI IV

```
        row1_col1.label(item.name)
        row2_col1.label("Version: "+str(item.version))
        row1_col2.label(item.format)
        row2_col2.label("Status: "+str(item.status))
        load = col2.operator("connector.load")
        load.item.name = item.name
        load.item.id = item.id
        load.item.version = item.version
        load.item.format = item.format
        load.item.model_id = item.model_id
        load.item.model_vers = item.model_vers


    layout.template_list("FileList", "",
scene.prop_group_files, "coll", scene.prop_group_files,
"index", rows=5, maxrows=5, columns=3, type='DEFAULT')
```

Co je třeba řešit u zásuvných modulů

- Typ modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- **Použité matematické a grafické funkce**
- API rozšiřované aplikace

Použité matematické a grafické funkce

- math
- mathutils
  - Matematické typy (vektor, matice, barva, atd)
  - Operace nad vektory a maticemi (rotace, determinat, atd.)
  - interpolátory
  - stromové struktury
- SciPy
- NumPy

Ostatní

- request
- json
- time
- random

Co je třeba řešit u zásuvných modulů

- Typ modulu
- Nastavení prostředků (konfigurace)
- Registrace
- Uživatelské rozhraní modulu
- Použité matematické a grafické funkce
- **API rozšiřované aplikace**

API rozšiřované aplikace

- Context Access (bpy.context)
  - režimy Blenderu a informace o scéně
  - informace o výběru
  - nastavení Blenderu
- Data Access (bpy.data)
  - přístup k datům
  - bpy.data.meshes["Cube"]
- Operators (bpy.ops)
  - volání funkcí blenderu
  - bpy.ops.mesh.subdivide(number_cuts=3, smoothness=0.5)
- Types (bpy.types)
  - struktury elementů scény
  - konstanty

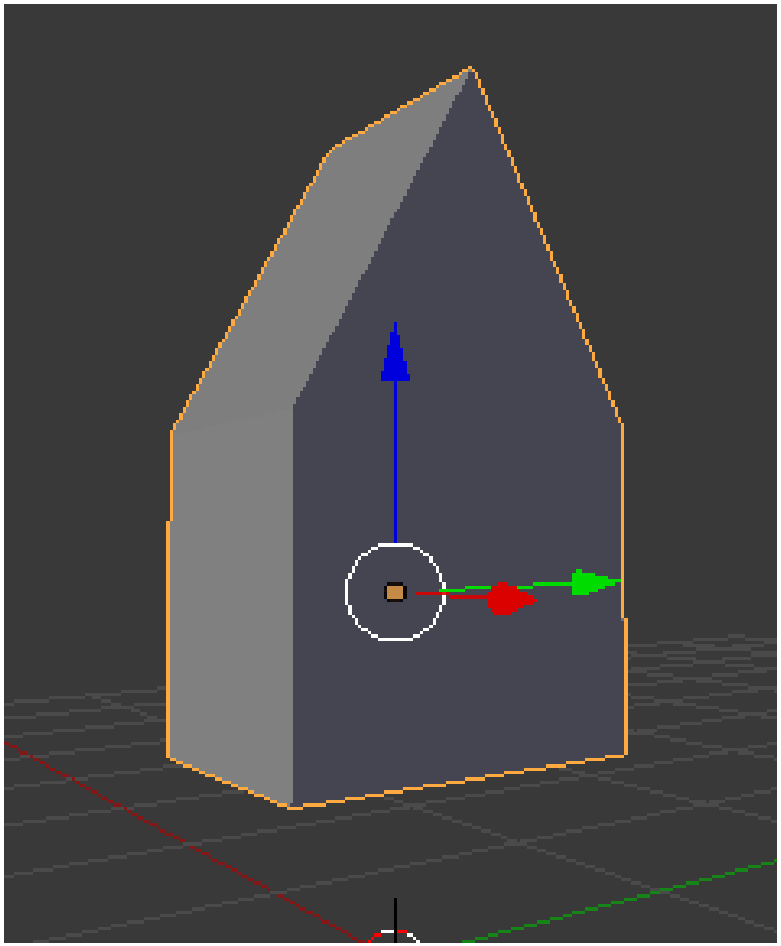
API rozšiřované aplikace

- Utilities (bpy.utils)
  - pomocné fukce
  - registrace modulů
  - preview - bpy.utils submodule (bpy.utils.previews)
- Path Utilities (bpy.path)
  - správa adresářové struktury v Blenderu
- Application Data (bpy.app)
  - aplikační nastavení
  - události - Application Handlers (bpy.app.handlers)
  - lokalizace - Application Translations (bpy.app.translations)
- Property Definitions (bpy.props)
  - UI (bool, int float, enum, collection, vector)
  - management parametrů

API rozšiřované aplikace

```
from bpy.props import *
    def create_mesh_object(context, verts, edges,
faces, name):
    # Vytvoření nové a prázdné mesh
    mesh = bpy.data.meshes.new(name)
    # konfigurace meshe prostřednictvím
verts/edges/faces.
    mesh.from_pydata(verts, edges, faces)
    # Aktualizace mesh.
    mesh.update()
    from bpy_extras import object_utils
    return object_utils.object_data_add(context, mesh,
operator=None)
```

Příklad



Příklad

```
def add_house(size = (1. ,1. ,1.), strecha = 1):
    verts = []
    edges = []
    faces = []

    size_x = size[0] / 2.0
    size_y = size[1] / 2.0
    size_z = size[2] / 2.0
```

Příklad

```
verts.append((-size_x, -size_y, -size_z))
verts.append(( size_x, -size_y, -size_z))
verts.append(( size_x, size_y, -size_z))
verts.append((-size_x, size_y, -size_z))
verts.append((-size_x, -size_y, size_z))
verts.append(( size_x, -size_y, size_z))
verts.append(( size_x, size_y, size_z))
verts.append((-size_x, size_y, size_z))
verts.append((-size_x, 0, size_z+strecha))
verts.append(( size_x, 0, size_z+strecha))
```

Příklad

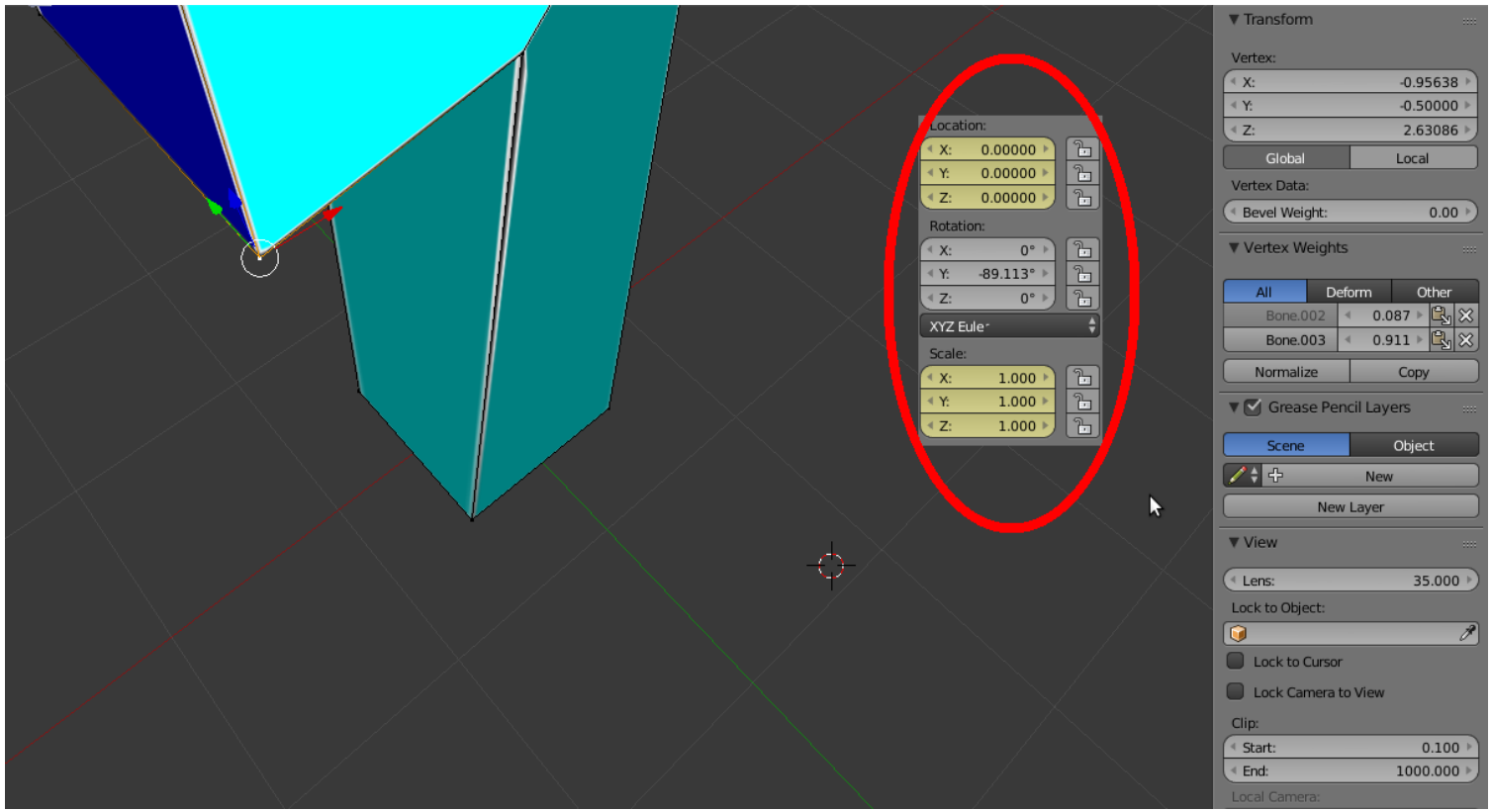
```
faces.append([0,1,2,3])
faces.append([0,1,5,4])
faces.append([1,2,6,5])
faces.append([2,3,7,6])
faces.append([3,0,4,7])

faces.append([4,5,9,8])
faces.append([6,7,8,9])
faces.append([5,6,8])
faces.append([4,7,9])
return verts, edges, faces
```

Vytváření geometrie a objektů - Geometrie (Mesh)

Základem je objekt Mesh

- Základní getry
  - vertices, edges, loops polygons, skin_vertices
  - total_vert_sel, total_edge_sel, total_face_sel
  - is_editmode
  - shape_keys
- Vytvoření
  - from_pydata
- Manipulace
  - transform
- Validace a aktualizace
  - validate
  - update

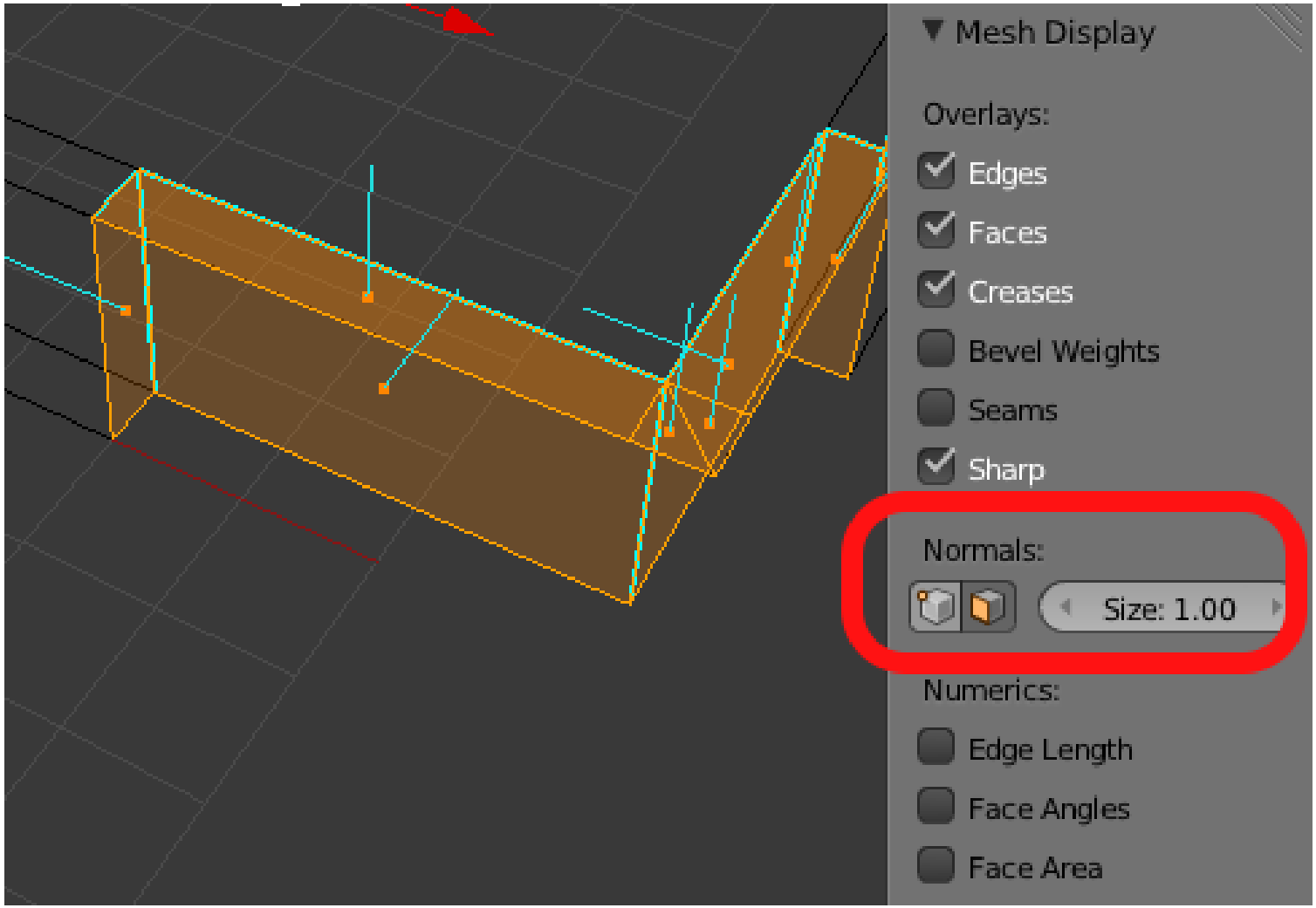
Vytváření geometrie a objektů - Geometrie (Mesh)



Vytváření geometrie a objektů - Geometrie (Mesh)

- Nastavení zobrazení
  - show_edges, show_faces
  - show_edge_seams
  - show_normal_vertex, show_normal_face
  - show_weight
- Výpočty spojení a rozdělení
  - calc_normals
  - calc_normals_split
  - calc_smooth_groups
  - free_normals_split

Vytváření geometrie a objektů - Geometrie (Mesh)

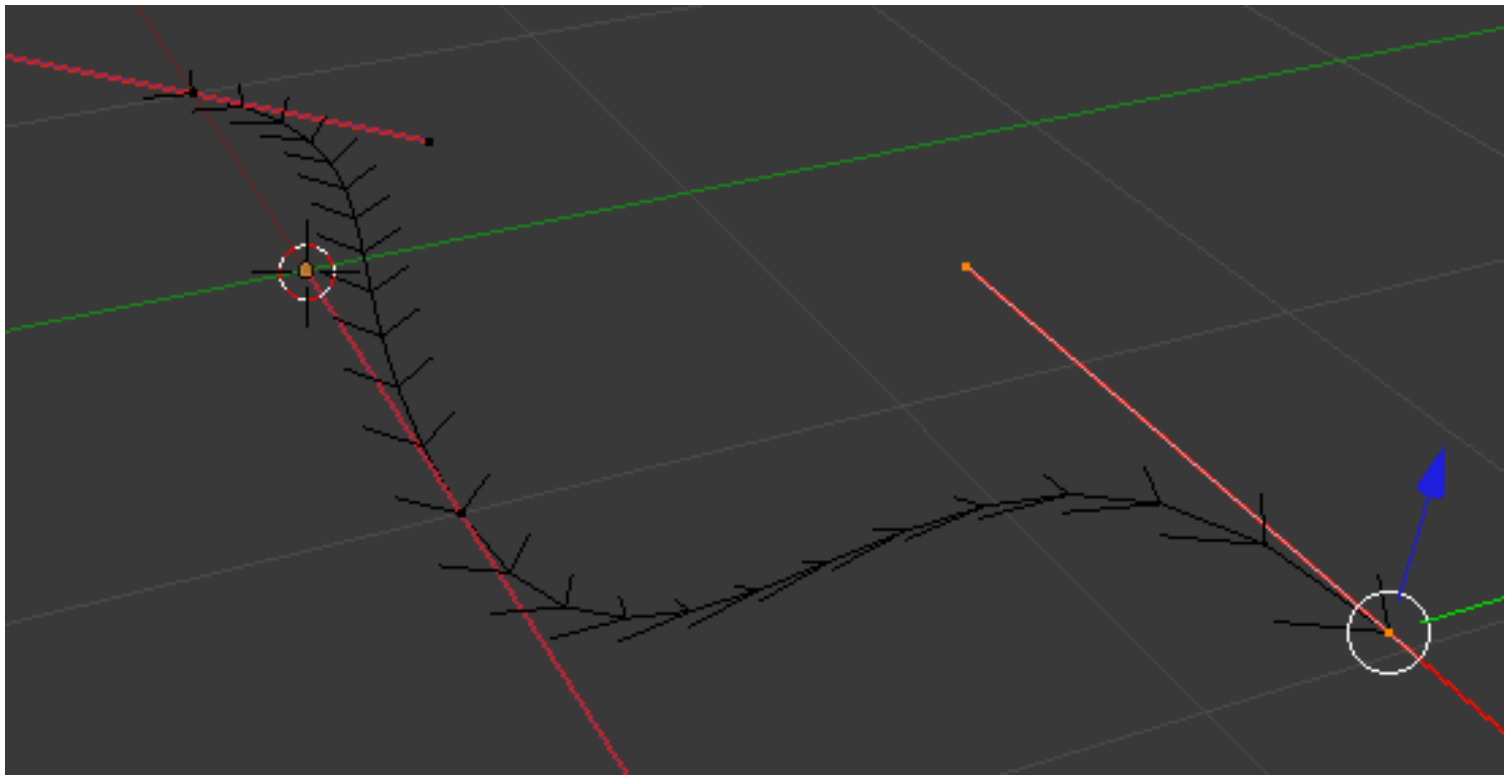


Vytváření geometrie a objektů - Křivky

Základem je objekt Curve

- Manipulace
  - primitive_bezier_circle_add, primitive_bezier_curve_add (obdobně pro NURBS)
  - duplicate, extrude, subdivide
  - delete, separate, split
  - make_segment,
  - radius_set
- Výběry
  - de_select_first, de_select_last, select_previous, select_next
  - hide
  - select_all, select_linked, select_row
  - select_random

Vytváření geometrie a objektů - Křivky



Vytváření geometrie a objektů - Křivky příklad

```
bl_info = {
    "name": "Curly Curves",
    "description": "Zjednodušený příklad",
    "author": "Cmomoney",
    "version": (1, 0, 0),
    "blender": (2, 69, 0),
    "location": "View3D > Add > Curve > Curly Curve",
    "warning": "",
    "wiki_url": "http://adresa .addonu",
    "tracker_url":
"https://projects.blender.org/tracker/index.php?func=deta
    "category": "Add Curve"
}
```

Vytváření geometrie a objektů - Křivky příklad

```
import bpy
from bpy.types import Operator
from bpy.props import *
from bpy_extras.object_utils import AddObjectHelper,
object_data_add
from mathutils import Vector
```

Vytváření geometrie a objektů - Křivky příklad

```
def add_type(self, context):
    s_x = self.scale_x
    s_y = self.scale_y
    verts = [[-0.71753 * s_x, -0.087 * s_y, 0,...]]
    lhan = [[(-0.818 * s_x, -0.143 * s_y, 0), ...)]]
    rhan = [[(-0.616 * s_x, -0.032 * s_y, 0), ...]]
    make_curve(self, context, verts, lhan, rhan)
```

Vytváření geometrie a objektů - Křivky příklad

```
def make_curve(self, context, vs, lh, rh):
    scale_x = self.scale_x
    scale_y = self.scale_y
    type = self.type
    curve_data = bpy.data.curves.new(name='CurlyCurve',
type='CURVE')
    curve_data.dimensions = '3D'
```

Vytváření geometrie a objektů - Křivky příklad

```
for p in range(len(vs)):
    c = 0
    spline = curve_data.splines.new(type='BEZIER')
    spline.bezier_points.add( len(vs[p])/3-1 )
    spline.bezier_points.foreach_set('co', vs[p])
    for bp in spline.bezier_points:
        bp.handle_left_type = 'ALIGNED'
        bp.handle_right_type = 'ALIGNED'
        bp.handle_left.xyz = lh[p][c]
        bp.handle_right.xyz = rh[p][c]
        c += 1
    spline.bezier_points[3].handle_left.xyz = lh[p][3]
object_data_add(context, curve_data, operator=self)
```

Vytváření geometrie a objektů - Křivky příklad

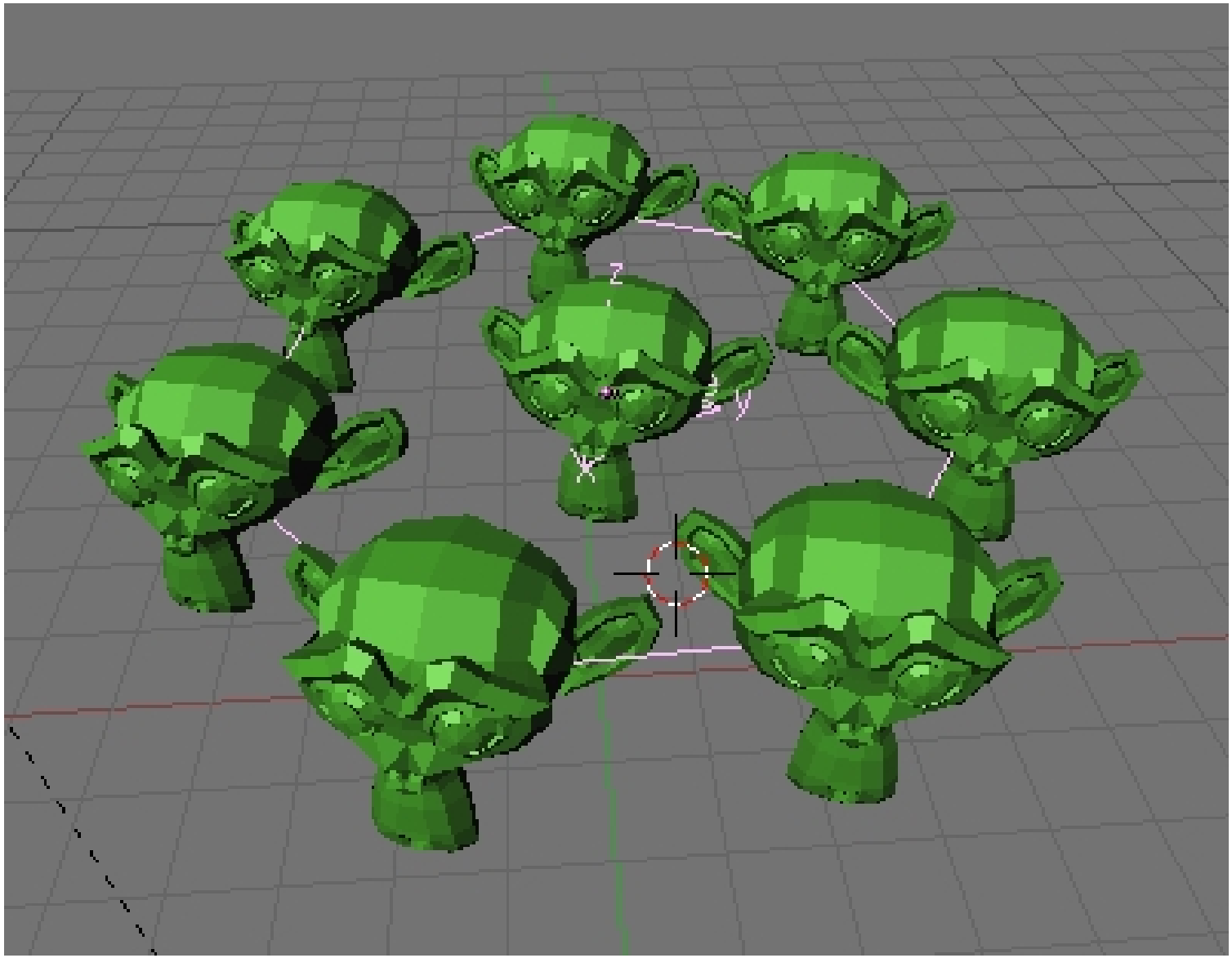
```
class add_curlycurve(Operator, AddObjectHelper):
    bl_idname = "curve.curlycurve"
    bl_label = "Add Curly Curve"
    bl_options = {'REGISTER', 'UNDO'}
    scale_x = FloatProperty(name="scale x",
description="scale on x axis", default=1.0)
    scale_y = FloatProperty(name="scale y",
description="scale on y axis", default=1.0)
    def execute(self, context):
        add_type(self, context)
        return {'FINISHED'}
```

Vytváření geometrie a objektů - Objekty

Základem je objekt Object

- Vytvoření a mazání
  - add, add_named
  - duplicate, duplicate_move_linked
  - join, join_shapes
  - delete
- Střed
  - origin_clear, origin_set

Vytváření geometrie a objektů - Objekty

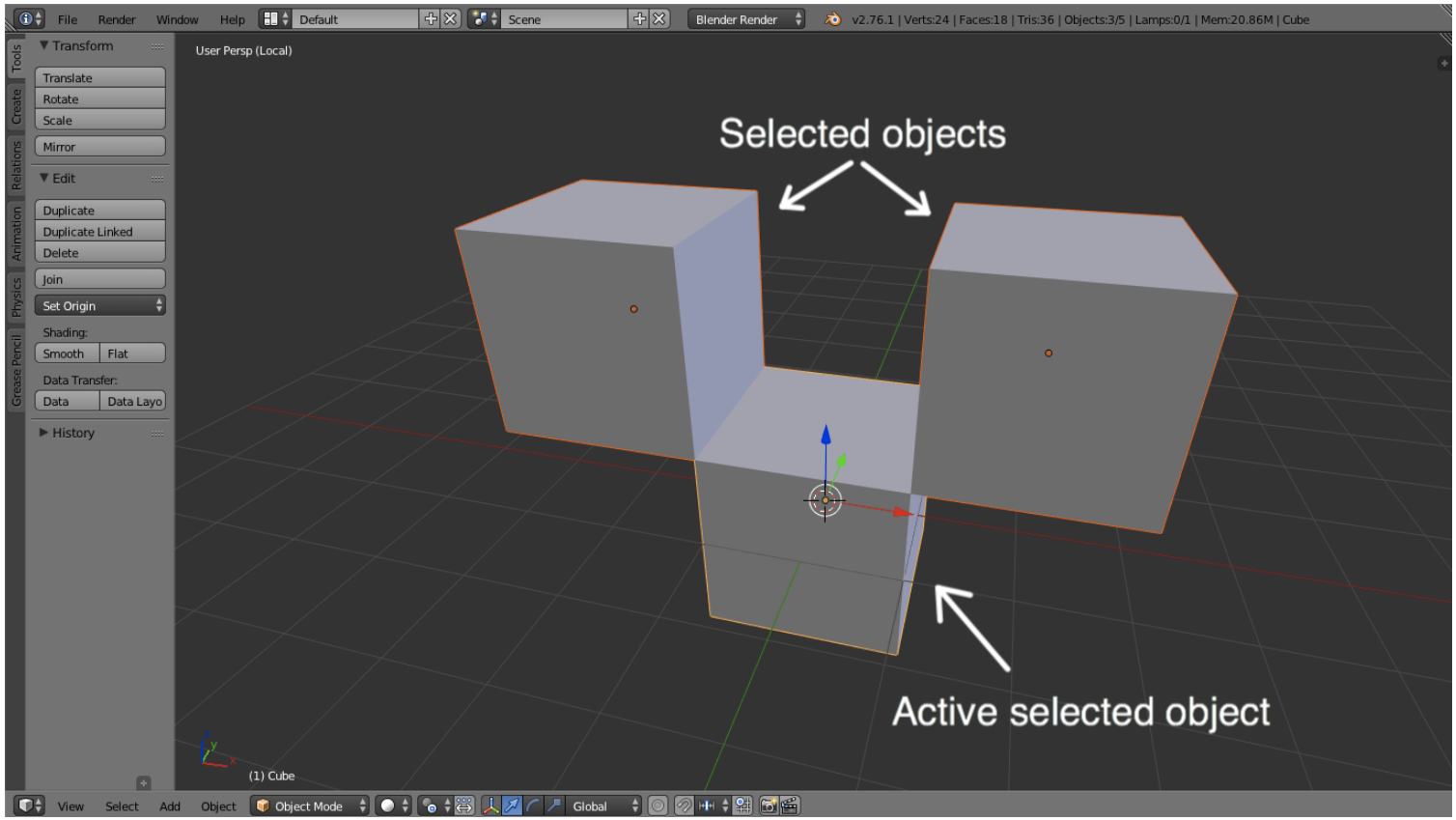


Manipulace s objekty

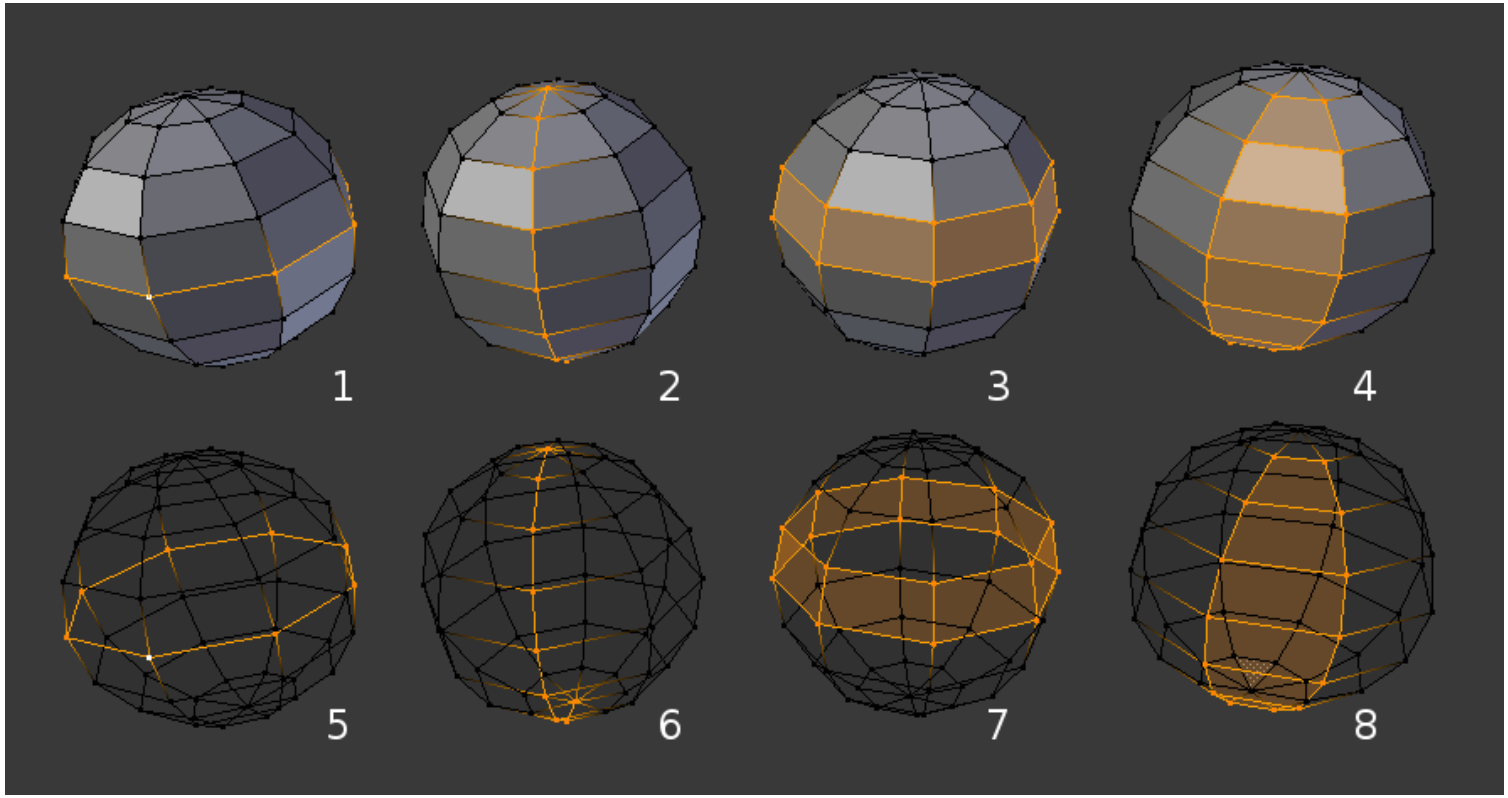
Základem je objekt Object

- Transformace
  - transform_apply, location_clear
  - rotation_clear, location_clear, scale_clear
  - convert
  - group_add, group_link, group_remove
  - modifier_add, modifier_apply, modifier_convert. modifier_copy, modifier_remove
  - parent_set
- Výběry
  - select_all
  - select_by_layer, select_by_type
  - select_grouped, select_linked
  - select_pattern

Vytváření geometrie a objektů - Objekty



Vytváření geometrie a objektů - Geometrie (Mesh)

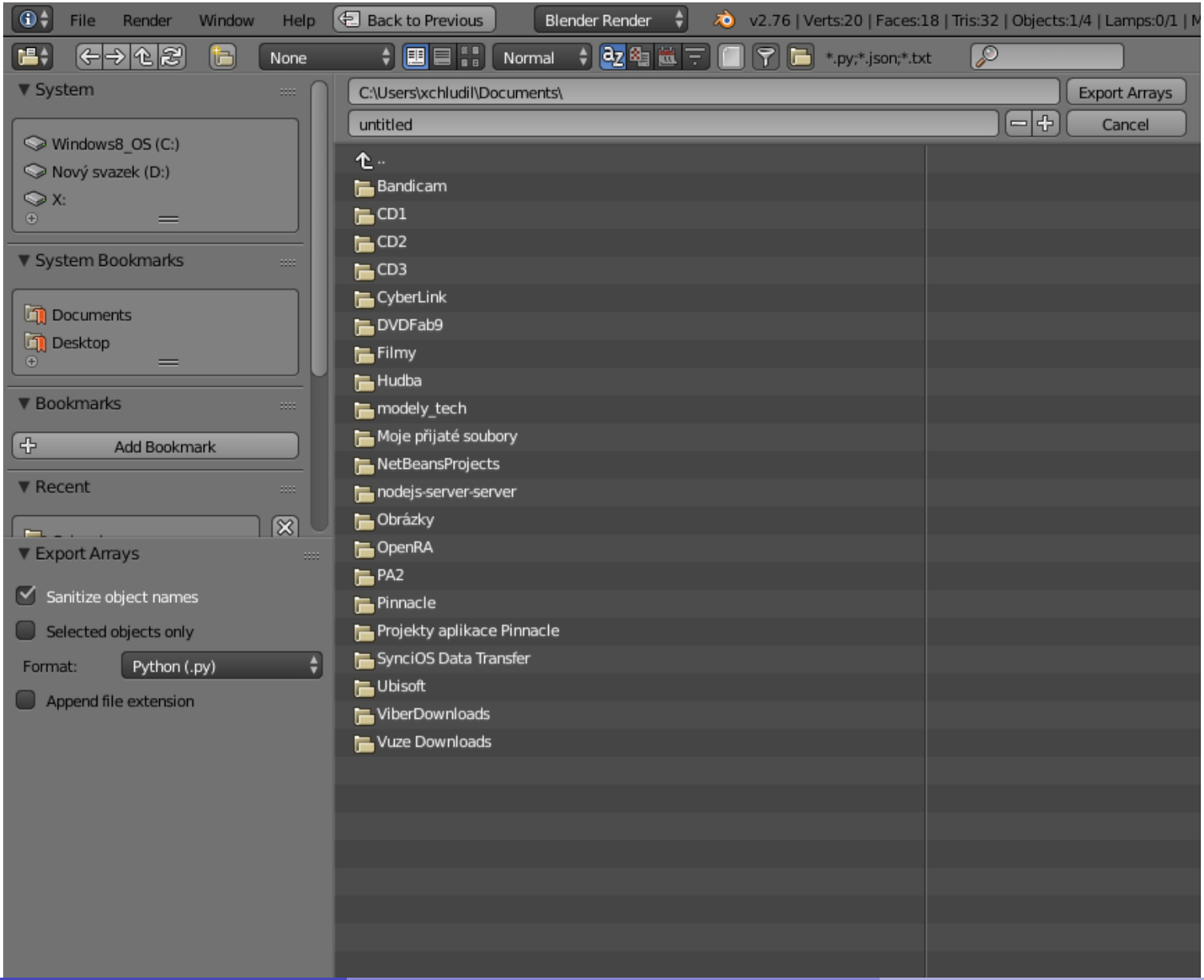


Import a Export

Základem je práce se souborem a získání nastavení z GUI Blenderu

- Serializace
  - Vytvoření obsahu ve formě objektu
  - formátovače (json.dumps, repr)
- Práce se souborem, nastavení GUI Blenderu
  - definice atributu (selected_only, format, extensions)
  - open, write, read

Import a Export

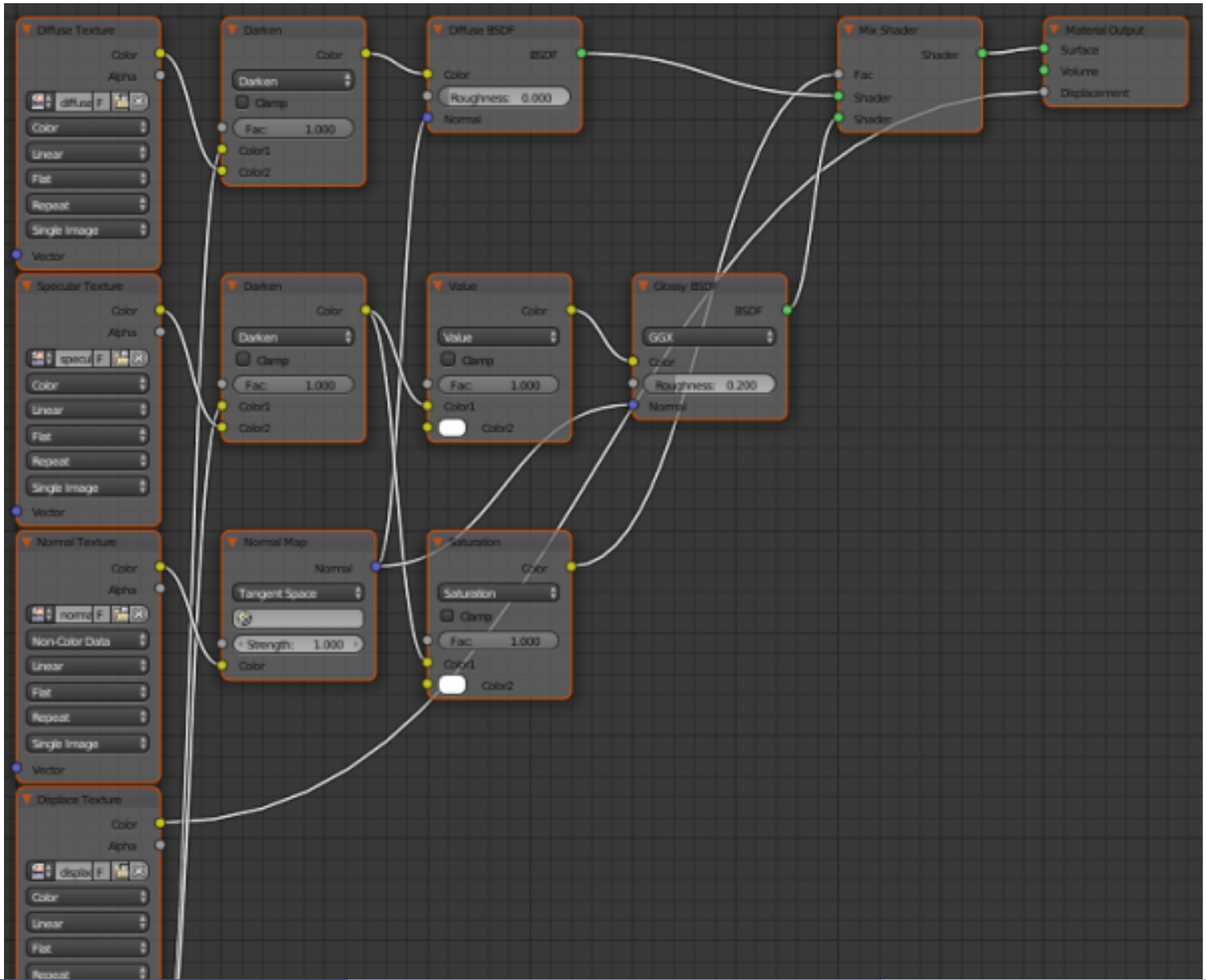


Materiály a textury - Materiály

Základem je objekt Object a Material

- Object
  - material_slot_add, material_slot_copy(), material_slot_assign(), ...
- Material
  - new, copy, paste, ..
  - material.node_tree
  - node (location, color, label ...)
  - NodeLinks (new, remove, clear)

Materiály a textury - Materiály

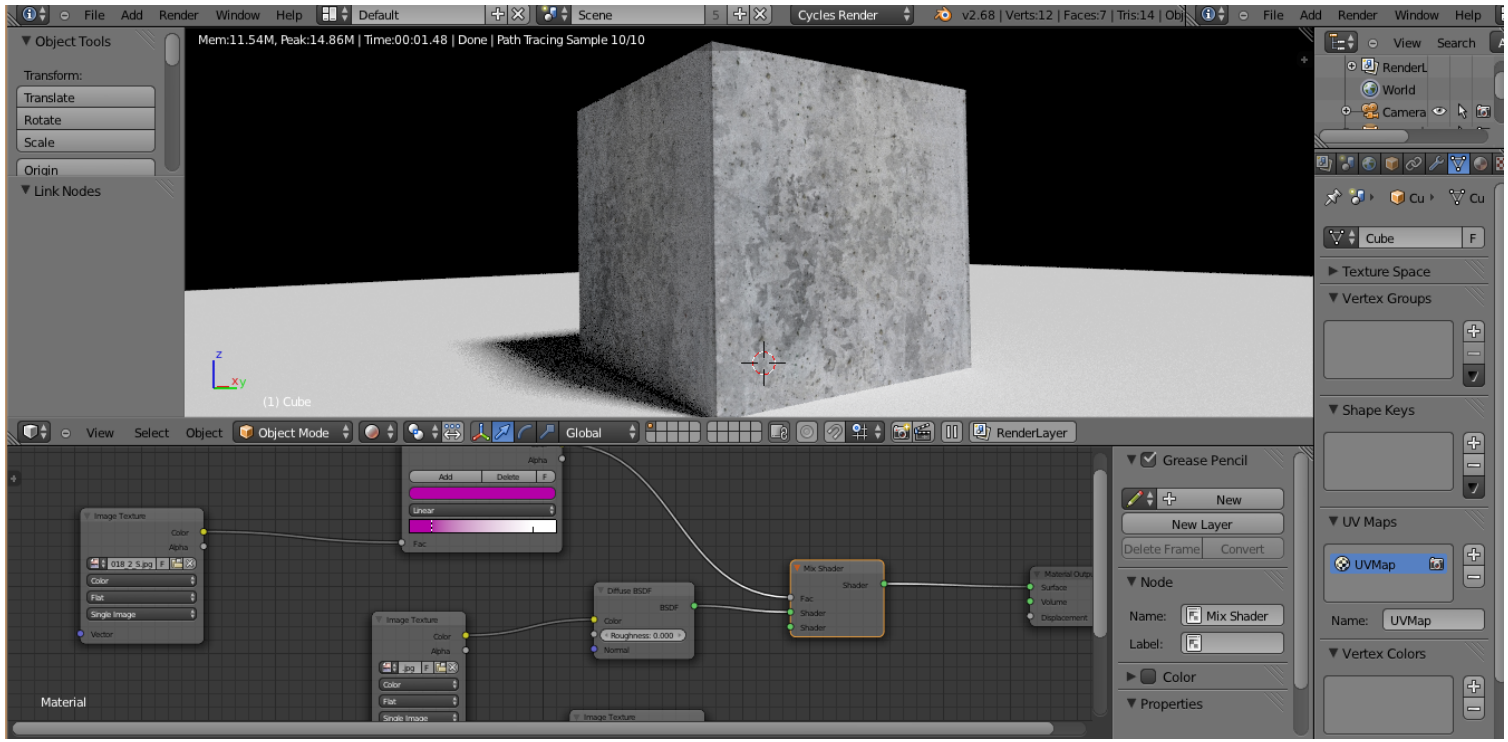


Materiály a textury - Mapování, Textury

Vhodné řešení je využít Cycles, vše jsou uzly

- Texture
  - new, copy, paste, ..
  - node_tree
  - node (location, color, label ...)
  - NodeLinks (new, remove, clear)

Materiály a textury - Mapování, Textury



Co bychom po dnešku měli znát

Témata probraná na dnešní přednášce:

- Základní struktory Blender API
- Funkce podporující různé typy modulů
- Základy UI