Washington

October 12, 2021

[1]:
```
%load_ext autoreload
%autoreload 2
import sys; sys.path.append('../')

# Prettymaps
from prettymaps import *
# Vsketch
import vsketch
# OSMNX
import osmnx as ox
# Matplotlib-related
import matplotlib.font_manager as fm
from matplotlib import pyplot as plt
from descartes import PolygonPatch
# Shapely
from shapely.geometry import *
from shapely.affinity import *
from shapely.ops import unary_union
```

[ ]:
```
# Style parameters
palette = ['#FFC857', '#E9724C', '#C5283D']
background_c = '#F2F4CB'
dilate = 300

# Setup figure
fig, ax = plt.subplots(figsize = (10, 10), constrained_layout = True)

# Plot
layers = plot(
    (38.89872,-77.03654),
    radius = 1150,
    ax = ax,
    layers = {
        'perimeter': {'circle': False, 'dilate': dilate},
        'streets': {
            'width': {
                    'motorway': 5,
```

```
                'trunk': 5,
                'primary': 4.5,
                'secondary': 4,
                'tertiary': 3.5,
                'residential': 3,
                'service': 2,
                'unclassified': 2,
                'pedestrian': 2,
                'footway': 1,
            },
            'circle': False,
            'dilate': dilate
        },
        'building': {
            'tags': {
                'building': True,
                'landuse': 'construction'
            },
            'union': False,
            'circle': False,
            'dilate': dilate
        },
        'water': {
            'tags': {
                'natural': ['water', 'bay']
            },
            'union': False,
            'circle': False,
            'dilate': dilate
        },
        'green': {
            'tags': {
                'landuse': ['grass', 'natrual'],
                'natural': ['island', 'wood'],
                'leisure': 'park'
            },
            'circle': False,
            'dilate': dilate
        },
        'forest': {
            'tags': {
                'landuse': 'forest'
            },
            'circle': False,
            'dilate': dilate
        },
        'parking': {
```

```
            'tags': {
                'amenity': 'parking', 'highway': 'pedestrian', 'man_made':␣
 ↪'pier'
            },
            'circle': False,
            'dilate': dilate
        },
    },

            drawing_kwargs = {
            'background': {'fc': '#F2F4CB', 'ec': '#dadbc1', 'hatch': 'ooo...',␣
 ↪'zorder': -1},
            'perimeter': {'fill': False, 'lw': 0, 'zorder': 0},
            'green': {'fc': '#D0F1BF', 'ec': '#2F3737', 'lw': 1, 'zorder': 1},
            'forest': {'fc': '#64B96A', 'ec': '#2F3737', 'lw': 1, 'zorder': 1},
            'water': {'fc': '#a1e3ff', 'ec': '#2F3737', 'hatch': 'ooo...',␣
 ↪'hatch_c': '#85c9e6', 'lw': 1, 'zorder': 2},
            'parking': {'fc': '#F2F4CB', 'ec': '#2F3737', 'lw': 1, 'zorder': 3},
            'streets': {'fc': '#2F3737', 'ec': '#475657', 'alpha': 1, 'lw': 0,␣
 ↪'zorder': 3},
            'building': {'palette': palette, 'ec': '#2F3737', 'lw': .5,␣
 ↪'zorder': 4},

        },

    osm_credit = {'x': .02, 'y': .05, 'color': '#2F3737'}
)

# Set bounds
xmin, ymin, xmax, ymax = layers['perimeter'].bounds
dx, dy = xmax-xmin, ymax-ymin
ax.set_xlim(xmin-.06*dx, xmax+.06*dx)
ax.set_ylim(ymin-.06*dy, ymax+.06*dy)

# Draw left text
#ax.text(
 #   xmin-.06*dx, ymin+.5*dy,
  #  'Barcelona, Spain',
   # color = '#2F3737',
   # rotation = 90,
   # fontproperties = fm.FontProperties(fname = '../assets/Permanent_Marker/
 ↪PermanentMarker-Regular.ttf', size = 35),
#)
# Draw top text
ax.text(
    xmin+0*dx, ymin-.05*dy,
```

```
    "White House, Washington",
    color = '#2F3737',
    fontproperties = fm.FontProperties(fname = '../assets/Permanent_Marker/
 ↪PermanentMarker-Regular.ttf', size = 35),
)

plt.savefig('prints/washington.png')
plt.savefig('prints/washington.svg')
```

```
/opt/conda/lib/python3.9/site-packages/ipykernel/ipkernel.py:283:
DeprecationWarning: `should_run_async` will not call `transform_cell`
automatically in the future. Please pass the result to `transformed_cell`
argument and any exception that happen during thetransform in
`preprocessing_exc_tuple` in IPython 7.17 and above.
  and should_run_async(code)
```

[ ]: