Autograph: Basic

In this ungraded lab, you will go through some of the basics of autograph so you can explore what the generated code looks like.

Imports

In [1]:
```
import tensorflow as tf
```

Addition in autograph

You can use the `@tf.function` decorator to automatically generate the graph-style code as shown below:

In [2]:
```
@tf.function
def add(a, b):
    return a + b


a = tf.Variable([[1.,2.],[3.,4.]])
b = tf.Variable([[4.,0.],[1.,5.]])
print(tf.add(a, b))

# See what the generated code looks like
print(tf.autograph.to_code(add.python_function))
```

```
tf.Tensor(
[[5. 2.]
 [4. 9.]], shape=(2, 2), dtype=float32)
def tf__add(a, b):
    with ag__.FunctionScope('add', 'fscope', ag__.ConversionOptions(recursive=True, user_requested=True, optional_features=
(), internal_convert_user_code=True)) as fscope:
        do_return = False
        retval_ = ag__.UndefinedReturnValue()
        try:
            do_return = True
            retval_ = (ag__.ld(a) + ag__.ld(b))
        except:
            do_return = False
            raise
        return fscope.ret(retval_, do_return)
```

if-statements in autograph

Control flow statements which are very intuitive to write in eager mode can look very complex in graph mode. You can see that in the next examples: first a simple function, then a more complicated one that involves lots of ops and conditionals (fizzbuzz).

In [3]:
```
# simple function that returns the square if the input is greater than zero
@tf.function
def f(x):
    if x>0:
        x = x * x
    return x

print(tf.autograph.to_code(f.python_function))
```

```
def tf__f(x):
    with ag__.FunctionScope('f', 'fscope', ag__.ConversionOptions(recursive=True, user_requested=True, optional_features=(),
internal_convert_user_code=True)) as fscope:
        do_return = False
        retval_ = ag__.UndefinedReturnValue()

        def get_state():
            return (x,)

        def set_state(vars_):
            nonlocal x
            (x,) = vars_

        def if_body():
            nonlocal x
            x = (ag__.ld(x) * ag__.ld(x))

        def else_body():
            nonlocal x
            pass
        ag__.if_stmt((ag__.ld(x) > 0), if_body, else_body, get_state, set_state, ('x',), 1)
        try:
            do_return = True
            retval_ = ag__.ld(x)
        except:
            do_return = False
            raise
        return fscope.ret(retval_, do_return)
```

Fizzbuzz in autograph

You may remember implementing fizzbuzz in preparation for a coding interview.

- Imagine how much fun it would be if you were asked to impement the graph mode version of that code!

Fortunately, you can just use `@tf.function` and then call `tf.autograph.to_code`!

In [4]:
```
@tf.function
def fizzbuzz(max_num):
    counter = 0
    for num in range(max_num):
        if num % 3 == 0 and num % 5 == 0:
            print('FizzBuzz')
        elif num % 3 == 0:
            print('Fizz')
        elif num % 5 == 0:
            print('Buzz')
        else:
            print(num)
        counter += 1
    return counter

print(tf.autograph.to_code(fizzbuzz.python_function))
```

```
def tf__fizzbuzz(max_num):
    with ag__.FunctionScope('fizzbuzz', 'fscope', ag__.ConversionOptions(recursive=True, user_requested=True, optional_featu
res=(), internal_convert_user_code=True)) as fscope:
        do_return = False
        retval_ = ag__.UndefinedReturnValue()
        counter = 0

        def get_state_3():
            return (counter,)

        def set_state_3(vars_):
            nonlocal counter
            (counter,) = vars_

        def loop_body(itr):
            nonlocal counter
            num = itr

            def get_state_2():
                return ()

            def set_state_2(block_vars):
                pass

            def if_body_2():
                ag__.ld(print)('FizzBuzz')

            def else_body_2():

                def get_state_1():
                    return ()

                def set_state_1(block_vars):
                    pass

                def if_body_1():
                    ag__.ld(print)('Fizz')

                def else_body_1():

                    def get_state():
                        return ()

                    def set_state(block_vars):
                        pass

                    def if_body():
                        ag__.ld(print)('Buzz')

                    def else_body():
                        ag__.ld(print)(ag__.ld(num))
                    ag__.if_stmt(((ag__.ld(num) % 5) == 0), if_body, else_body, get_state, set_state, (), 0)
                ag__.if_stmt(((ag__.ld(num) % 3) == 0), if_body_1, else_body_1, get_state_1, set_state_1, (), 0)
            ag__.if_stmt(ag__.and_((lambda : ((ag__.ld(num) % 3) == 0)), (lambda : ((ag__.ld(num) % 5) == 0))), if_body_2, e
lse_body_2, get_state_2, set_state_2, (), 0)
            counter = ag__.ld(counter)
            counter += 1
        num = ag__.Undefined('num')
        ag__.for_stmt(ag__.converted_call(ag__.ld(range), (ag__.ld(max_num),), None, fscope), None, loop_body, get_state_3,
set_state_3, ('counter',), {'iterate_names': 'num'})
        try:
            do_return = True
            retval_ = ag__.ld(counter)
        except:
            do_return = False
            raise
        return fscope.ret(retval_, do_return)
```