In [1]:
```
%matplotlib inline
# Importing standard Qiskit libraries and configuring account
from qiskit import QuantumCircuit, execute, Aer, IBMQ
from qiskit.compiler import transpile, assemble
from qiskit.tools.jupyter import *
from qiskit.visualization import *
import qiskit as q
from qiskit.tools.monitor import job_monitor

# Loading your IBM Q account(s)
provider = IBMQ.load_account()
```

Test 1

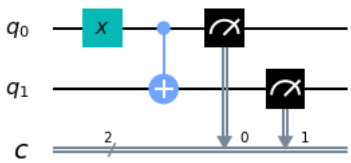
In [5]:
```
# 2 Quibits und 2 klassische Bits
circuit = QuantumCircuit(2,2)

# Einfügen eines NOT Gates
    # Bastelt aus 0,0 -> 1,0
circuit.x(0)

# Einfügen von controlled not (cnot).
    # Flipped das zweite Qbit wenn das erste eine 1 ist
circuit.cx(0, 1)

# Messen, Die Boxen zeigen an wie die Qubits in klassische Bits umgebaut werden
circuit.measure([0,1], [0,1])

circuit.draw()
```



In [9]:
```
# lassen wir das ma aufm Quantencomputer laufen!
# Erstma schauen, welcher Quantencomputer steht zur Verfügung, wieviele Qbits hat er und wieviele warten scho
n?

provider = IBMQ.get_provider("ibm-q")

for backend in provider.backends():
    try:
        qubit_count = len(backend.properties().qubits)
    except:
        qubit_count = "simulated"

    print(f"{backend.name()} has {backend.status().pending_jobs} qued and {qubit_count} qubits")
```

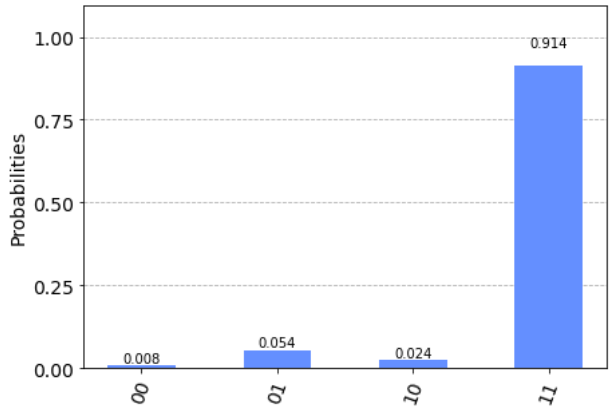
```
ibmq_qasm_simulator has 1 qued and simulated qubits
ibmqx2 has 2 qued and 5 qubits
ibmq_16_melbourne has 10 qued and 15 qubits
ibmq_vigo has 6 qued and 5 qubits
ibmq_ourense has 1 qued and 5 qubits
ibmq_london has 7 qued and 5 qubits
ibmq_burlington has 2 qued and 5 qubits
ibmq_essex has 5 qued and 5 qubits
ibmq_armonk has 0 qued and 1 qubits
ibmq_rome has 6 qued and 5 qubits
```

In [11]:
```
backend = provider.get_backend("ibmqx2")
job = execute(circuit, backend = backend, shots = 500)
job_monitor(job)
```

```
Job Status: job has successfully run
```

In [12]:
```
# Nachdem wirs ausgeführt haben (...WHAT THE FUCK????) plotten wir das ergebnis
result = job.result()
counts = result.get_counts(circuit)

plot_histogram([counts])
```



Test 2

Was können wir jetzt damit machen? Zum Testen nutzte ich aber jetzt die Simulation, das geht schneller

In [ ]:
```
def do_Job(circuit):
    # Ausführen und Plotten
    backend = provider.get_backend("ibmq_qasm_simulator")
    job = execute(NeuesCircuit, backend = backend, shots = 500)
    job_monitor(job)

    result = job.result()
    statevec = reult.get_statecover()

    counts = result.get_counts(circuit)

    return statevec, counts
```

In [ ]:
```
NeuesCircuit = QuantumCircuit(2,2)

statevec, counts = do_Jop(NeuesCircuit)

plot_bloch_multivector(statevec)
plot_histogram([counts])
```