

# Wrapping up & Next Steps

---

# Why P4<sub>16</sub>?

---

- **Clearly defined semantics**
  - You can describe what your data plane program is doing
- **Expressive**
  - Supports a wide range of architectures through standard methodology
- **High-level, Target-independent**
  - Uses conventional constructs
  - Compiler manages the resources and deals with the hardware
- **Type-safe**
  - Enforces good software design practices and eliminates “stupid” bugs
- **Agility**
  - High-speed networking devices become as flexible as any software
- **Insight**
  - Freely mixing packet headers and intermediate results



# Things we covered

---

- **The P4 "world view"**
  - Protocol-Independent Packet Processing
  - Language/Architecture Separation
  - If you can interface with it, it can be used
- **Key data types**
- **Constructs for packet parsing**
  - State machine-style programming
- **Constructs for packet processing**
  - Actions, tables and controls
- **Packet deparsing**
- **Architectures & Programs**



# Things we didn't cover

---

- **Mechanisms for modularity**
  - Instantiating and invoking parsers or controls
- **Details of variable-length field processing**
  - Parsing and deparsing of options and TLVs
- **Architecture definition constructs**
  - How these “templated” definitions are created
- **Advanced features**
  - How to do learning, multicast, cloning, resubmitting
  - Header unions
- **Other architectures**
- **Control plane interface**





# The P4 Language Consortium

- Consortium of academic and industry members
- Open source, evolving, domain-specific language
- Permissive Apache license, code on GitHub today
- Membership is free: contributions are welcome
- Independent, set up as a California nonprofit

**Protocol Independent**  
P4 programs specify how a switch processes packets.

**Target Independent**  
P4 is suitable for describing everything from high-performance forwarding ASICs to software switches.

**Field Reconfigurable**  
P4 allows network engineers to change the way their switches process packets after they are deployed.

```
table routing {
  reads {
    ipv4.dstAddr : lpm;
  }
  actions {
    do_drop;
    route_ipv4;
  }
  size: 2048;
}

control ingress {
  apply(routing);
}
```

**TRY IT** Get the code from P4factory



# Reception @ Riverwalk Bar & Grill

