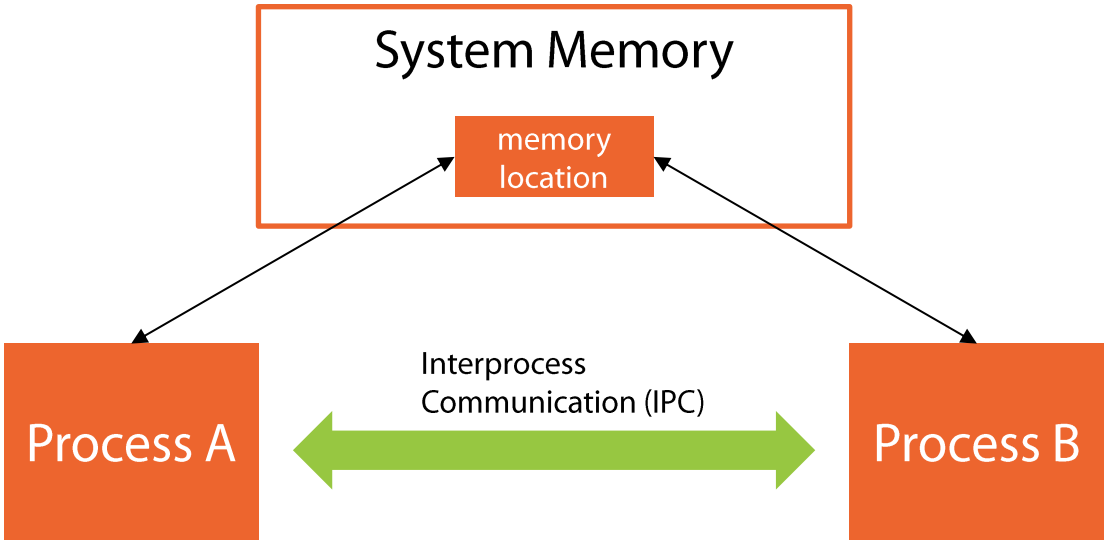
Interprocess Communication Using PipeStream

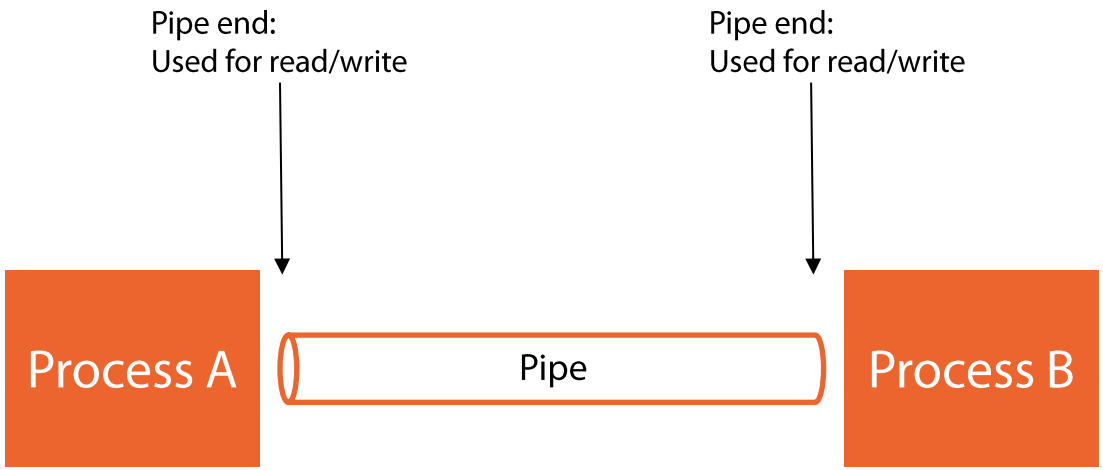
Mohamad Halabi
Microsoft Integration MVP
@mohamadhalabi

pluralsight
hardcore dev and IT training

Pipes


System Memory
memory location
Process A
Interprocess Communication (IPC)
Process B

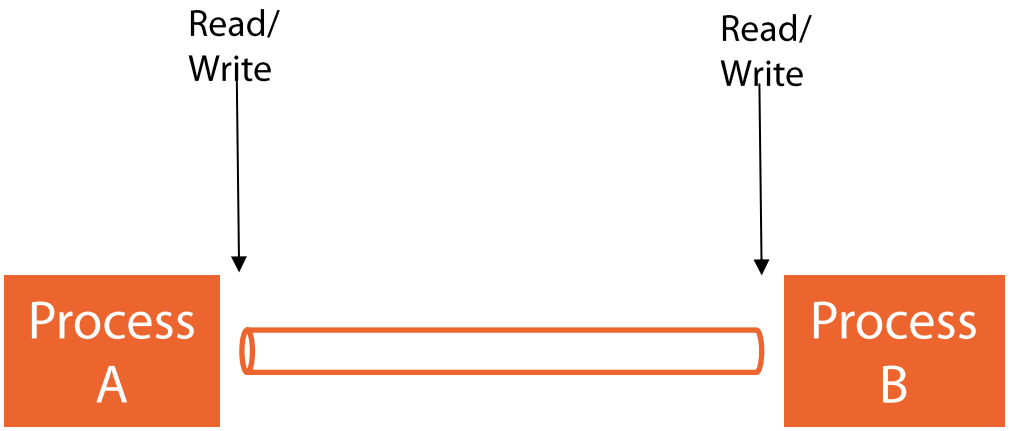
Pipes


Pipe end:
Used for read/write
Pipe end:
Used for read/write
Process A
Pipe
Process B

Communication Mode

- **One-way**

- **Duplex (Two-way)**



Pipe Server and Pipe Client

- One of the processes creates (instantiates) the pipe

Named and Anonymous Pipes

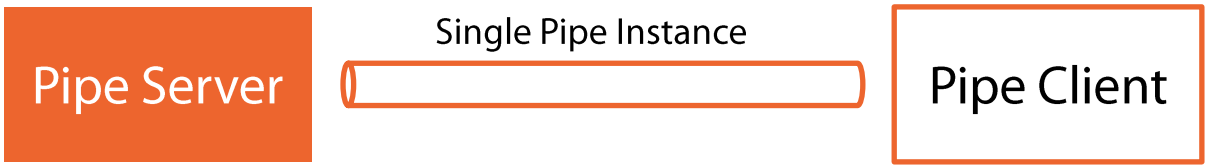
Named	Anonymous
Has a name	Has no name
One-way / Duplex	One-way
Pipe server with multiple pipe clients	Pipe server / pipe client communication
Communication over same/different machine(s)	Communication over same machine

Note: Named pipes are discussed going forward

Single vs. Cross-Machine Communication

- **Named pipes support cross-machine process communication**
- **Same machine communication yields the best performance**
  - No network dependency == better performance

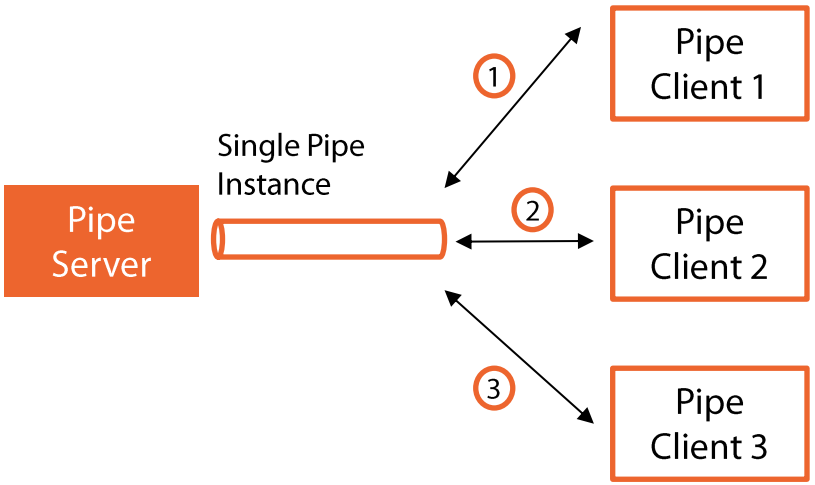
A Simple Pipe Server



A Named Pipe Server

- **Named pipes allow multiple client communication**

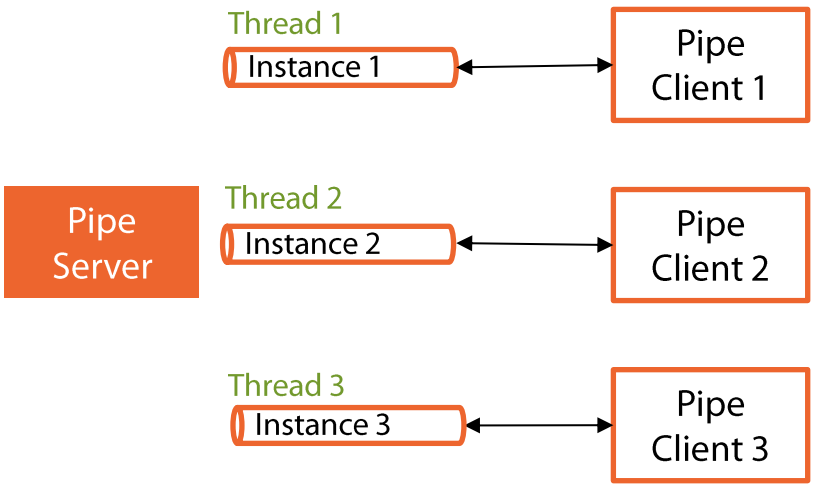
Option1: Single Pipe Instance



Cons: Easy implementation

Pros: Poor performance - sequential connectivity

Option2: Multiple Pipe Instances



Cons: Good performance

Pros: Handle server multithreading

PipeStream

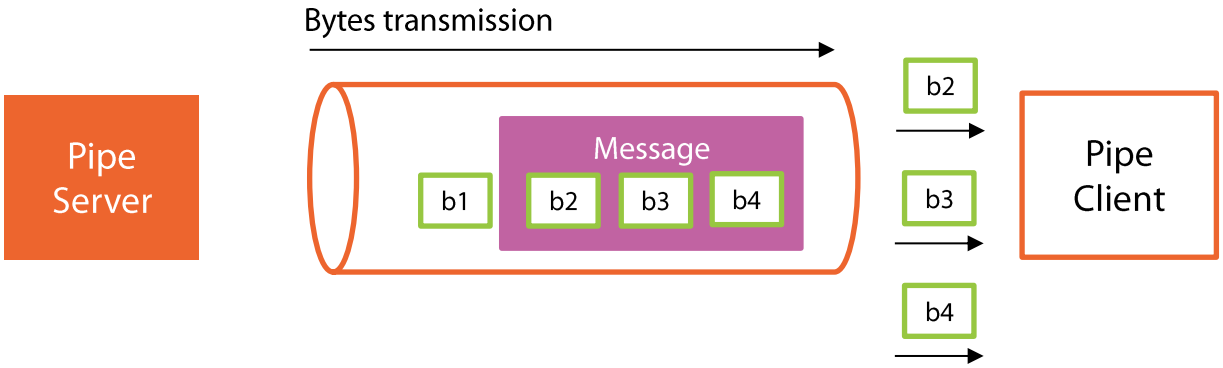
- **Pipes are stream-based**
  - The sending process sends data in a streamed fashion
  - The receiving process receives data in chunks as a series of bytes
- **System.IO.Pipes.PipeStream**
  - Not seekable
  - Four concrete implementations
    - AnonymousPipeServerStream and AnonymousPipeClientStream
    - NamedPipeServerStream and NamedPipeClientStream
  - Pipes as backing stores

Thread Safety

- **System.IO.Stream implementations – as a general rule – are not thread-safe**
  - Stream.Synchronized static method wraps stream with a thread-safe version
- **PipeStream is thread-safe**
  - No need to explicitly use Synchronized method
  - Read and write operations block the thread

Message Transmission

- **A Message is a collection (array) of bytes**
  - A Message is a grouping of related bytes



Message Transmission

- **For message transmission:**
  - Set PipeTransmissionMode enumeration to Message
  - Read and Write methods read/write messages (instead of bytes)
  - IsMessageComplete property of PipeStream indicates message completion

Summary

- **Pipes are used for interprocess communication**
- **A pipe is a shared memory between communicating processes**
- **A pipe server is the process that creates the pipe**
- **Named pipes:**
  - One-way / duplex
  - Multiple clients
  - Same machine and cross-machine communication
- **Anonymous pipes:**
  - One-way
  - Same machine communication

Summary

- **PipeStream abstract class supports stream-based data transmission**
  - AnonymousPipeServerStream and AnonymousPipeClientStream
  - NamedPipeServerStream and NamedPipeClientStream
- **Supports byte and message transmissions**