Quick Start

Get your first Pear P2P application running in less than 5 minutes.

What You'll Build

A simple peer-to-peer connection that exchanges messages between two running instances. This demonstrates the core of Pear: **connecting peers without servers**.

Prerequisites

- **Node.js 18+** - For package management only (Pear uses its own runtime)
- **5 minutes** - That's all you need!

Step 1: Install Pear

Open your terminal and run:

```bash
npm install -g pear
```

Verify the installation:

```bash
pear --version
```

▶ **Platform-Specific Requirements**

Step 2: Initialize Your Project

Create a new directory and initialize a Pear project:

```bash
mkdir my-first-p2p-app
cd my-first-p2p-app
pear init -y
```

This creates:

- `package.json` - Your project configuration
- `index.html` - The UI (for desktop apps)
- `app.js` - Your application logic

Step 3: Install Dependencies

Add the networking library:

```bash
npm install hyperswarm b4a
```

What these do:

- `hyperswarm` - Discovers and connects to peers
- `b4a` - Buffer utilities (JavaScript helper)

Step 4: Write Your Code

Replace the contents of `app.js` with:

```javascript
import Hyperswarm from 'hyperswarm'
import b4a from 'b4a'

// Create a swarm instance
const swarm = new Hyperswarm()

// Clean up when app closes
Pear.teardown(() => swarm.destroy())

// Generate a topic (or use one from command line)
const topicString = Pear.config.args[0] || 'my-first-app'
const topic = b4a.from(topicString, 'utf8')

console.log(`🍐 Joining topic: ${topicString}`)

// Join the swarm
const discovery = swarm.join(topic, { server: true, client: true })

// Wait for connections
swarm.on('connection', (peer) => {
  console.log('✅ Connected to a peer!')

  // Send a message
  peer.write('Hello from my Pear app!')

  // Receive messages
  peer.on('data', (data) => {
    console.log(`📨 Received: ${data.toString()}`)
  })
})

// Wait for the topic to be announced
discovery.flushed().then(() => {
  console.log('🔍 Searching for peers...')
  console.log('   Run this command in another terminal:')
```

```
  console.log(`   pear run --dev . ${topicString}`)
})
```

Step 5: Run Your App

Open **two terminals** side-by-side.

In **Terminal 1**, run:

```bash
pear run --dev .
```

You'll see:

```
🍐 Joining topic: my-first-app
🔍 Searching for peers...
   Run this command in another terminal:
   pear run --dev . my-first-app
```

In **Terminal 2**, run:

```bash
pear run --dev . my-first-app
```

🎉 **Success!** Both terminals will show:

```
✅ Connected to a peer!
📨 Received: Hello from my Pear app!
```

Understanding What Happened

Let's break down the code:

1. Creating a Swarm

```javascript
const swarm = new Hyperswarm()
```

This creates your P2P networking instance. It handles all the complexity of finding and connecting to peers.

2. Joining a Topic

```javascript
const topic = b4a.from('my-first-app', 'utf8')
swarm.join(topic, { server: true, client: true })
```

- **Topic**: A unique identifier that peers use to find each other
- **server: true**: Accept incoming connections
- **client: true**: Actively search for peers
- Think of it as a "room" that peers can join

3. Handling Connections

```javascript
swarm.on('connection', (peer) => {
  // This runs whenever a new peer connects
})
```

When two peers with the same topic find each other, they automatically connect.

4. Sending & Receiving Data

```javascript
peer.write('Hello!')           // Send data
peer.on('data', (data) => {    // Receive data
  console.log(data.toString())
})
```

Once connected, peers can exchange data like any network socket.

Try This Next

Now that you have a working P2P connection, try:

🔄 Connect Multiple Peers

Open 3, 4, or more terminals with the same topic. They'll all connect!

```bash
# Terminal 3
pear run --dev . my-first-app

# Terminal 4
pear run --dev . my-first-app
```

🌐 Connect Across Devices

Run the app on different computers on the internet. Use the same topic string and they'll find each other through the DHT (Distributed Hash Table).

💬 Add Interactive Chat

Modify the code to send messages from `stdin`:

```javascript
import process from 'bare-process'

process.stdin.on('data', (data) => {
  for (const peer of swarm.connections) {
    peer.write(data)
  }
})
```

Common Issues

▶ **My peers won't connect**

▶ **Error: Cannot find module**

▶ **Error: pear: command not found**

What You Learned

✅ How to install and set up Pear
✅ How to create a P2P connection using topics
✅ How to send and receive data between peers
✅ How Pear works without central servers

Next Steps

📚 Learn Core Concepts

Understand the fundamentals of P2P architecture Core Concepts →

🏗️ Build Your First App

Create a complete chat application with a UI
First App Tutorial →

🔍 Explore Examples

See what others have built with Pear
Examples Gallery →

📖 API Reference

Dive into the complete API documentation

API Docs →

Questions?

- **Not working?** → Troubleshooting Guide
- **Want to learn more?** → Core Concepts
- **Need help?** → Join our Discord

Edit this page on GitHub

Next page
Your First App