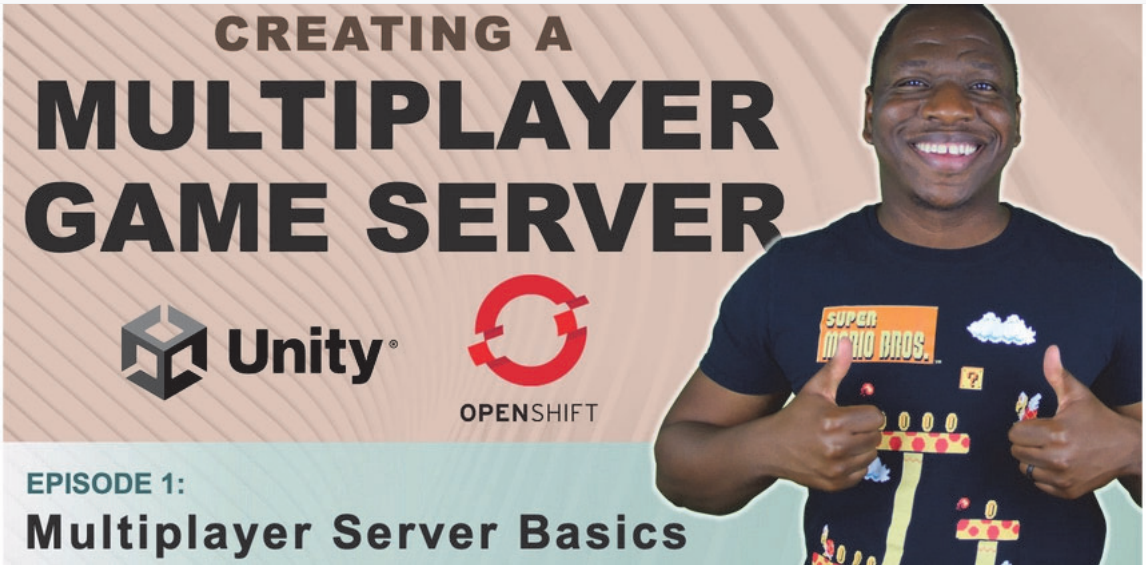


Bradston Henry for IBM Developer
Posted on Oct 25, 2021 • Edited on Nov 24, 2021

71 2

Multiplayer Server Basics | Creating a Multiplayer Game Server - Part 1

#gamedev #unity3d #cloud #redhat

Creating a Multiplayer Game Server (6 Part Series)

1. **Multiplayer Server Basics | Creating a Multiplayer Game Serv...**
2. Persistent vs Non-Persistent Connections | Creating a Multiplayer ...

... 2 more parts...

5. Creating Our Multiplayer Game using Unity | Creating a Multiplay...
6. Deploying Our Multiplayer Game Server to Red Hat OpenShift | C...

MDT Exhibit 2038
IPR2025-00542
Page 1

So as many of you know, I have been working in game development for many years (almost 10) and it is one of my favorite things to do. I have released multiple mobile games on the Android Playstore and have released and participated in game jams on the itch.io platform. But in all of my time doing game dev, I have always seen one aspect of game development, as out of my reach; Multiplayer Game Development.

But as I have grown as a developer/game developer in my professional career and in my personal dev skills, I have finally come to a place where I now know that I can actually accomplish that seemingly impossible feat.

And with that new found knowledge and confidence, I wanted to share with you all, the journey I took to get to this point and the skills that you will need to accomplish the same feat.

So, I am excited to start the "Creating a Multiplayer Game Server" series and share it with you all!

It will be a multiple episode/part series where I cover various topics that will culminate with YOU creating your own multiplayer game server that can be used in you own multiplayer game.

It will cover basic concepts about Multiplayer server and game design and the nitty-gritty of the code to make it happen.

Even if you are **not** a game dev, there is much you will be be able to learn from this series. I will be cover basic developer principles about server design and will be doing a deep dive into how to use Websocket technology to create continuous/persistent connections using the standard client-server model. I will also be covering topics like Red Hat OpenShift and how we can use it to deploy our multiplayer game server in the cloud.

My hope is that by the end of this series, anybody who desires will be able to create an online multiplayer game and will be able to understand what it takes to make one!

So without further adieu, Let's talk about episode one of the series:

"Multiplayer Server Basics"

MDT Exhibit 2038
IPR2025-00542
Page 2

Note: Since every episode/part of this series will have a corresponding Video, I will be doing a brief summary in my written blog of what was covered in the video.

Two Types of Multiplayer Game Servers

In order to start the process of building a multiplayer game it's crucial to understand the basics of what a multiplayer server is. So I will cover the two most common approaches to deploying multiplayer game servers in the real-world.

1. Local Player Hosted Game Servers

The local player hosted game server is an extremely common approach to implementing multiplayer into games in the gaming industry (e.g. Minecraft). Essentially, it allows for any particular player to host a multiplayer game session (that other players can join) on their local machine. In practice, a game developer enables their game to actually start a server on a players machine (that hosts the multiplayer game) and allows the player to broadcast to other players that their server is available to be joined and played on. In this way, any player anywhere in the the world can theoretically host a multiplayer game session that their friends can join.

So let's cover the pros and cons of using this multiplayer game server approach:

***PROS*:**

MDT Exhibit 2038
IPR2025-00542
Page 3

- **More Player Control Over Server**: This approach allows for players to have more discrete control over game sessions and servers. Players can implement "game mods" or "custom server implementations" depending on what the developer allows.
- **No Additional Operating Costs**: Since the player is hosting the server on their local machine (e.g. their personal computer) there is no additional costs associated to running this multiplayer server.
- **Easier to Scale-out**: Since developers do not have to manage how and when multiplayer servers are created, it is much easier for this type of multiplayer game server to scale. Developers need not be involved in the minutia of how many server may or may not be running to meet player demand.

***CONS*:**

- **Very Little Developer Control**: Since servers are managed on local players machines, it is difficult for developers to monitor or "police" player actions. This is generally okay for Co-operative games but can be a big issue for Competitive Games where cheating would easily run rampant.
- **Host Player Disconnection**: Since the game sessions are hosted and run on individual players local machines, if the hosting player were to disconnect from the gaming session for any reason (power outage, internet connectivity issues, etc.) the session could end for all players. There are some ways to mitigate this issue but it can be a very troublesome issue to design for.

2. Dedicated/Cloud Game Servers

The Dedicated/Cloud Game server approach is an also very common in the industry and used extremely regularly for Competitive based games (such Apex Legends and Call of Duty). The main feature of this approach is for game servers to be hosted somewhere on a remote server in the cloud. For example, the game server may be hosted in a Cloud platform such as Red Hat OpenShift and players join into game sessions that are hosted and created in that cloud. This approach takes the burden off of the average player for hosting and is primarily controlled and dictated by the developer of the game.

So let's cover the pros and cons of using the Dedicated/Cloud game server approach:

***PROS*:**

- **More Developer Control**: This approach allows for developers to primarily determine how players engage with multiplayer servers. This means that developers

MDT Exhibit 2038
IPR2025-00542
Page 4

can roll out updates that affect all players simultaneously and allows them to more easily manage the dynamics of competitive games where fairness (no cheating) is highly preferred.
- **Persistent Game States**: Because servers are always running and hosted somewhere in the cloud, this allows for players to be able to connect and play on the servers at anytime. This is great for games that have "shared-world" properties where game developers want worlds where players can change the state of the world and other players experience those changes at anytime.

***CONS*:**

- **More Demanding on Developer**: Since games are hosted on remote servers that players (generally) have no control over, it is important for developers to continually monitor and audit game servers. If there are any issues (such as server outages, buggy servers, etc) it is the developers responsibility to manage it. It also the developers responsibly to manage server scaling depending on player demand and player behaviour.
- **Additional Server Operating Costs**: Because the game servers need to be available to all players at all times, servers that are hosted and running, will incur some type of cost. In some cases, individual players (not developers) can host their own dedicated servers for games but they cannot avoid the cost that comes with this approach either.

So as you can see, there are plenty of pros and cons with either approach and it's important to understand those pros and cons when deciding the approach that is best for your game.

In this series, we will be tackling the Dedicated/Cloud Server approach. This is probably the most "complex" of the two approaches and I really would like to share with you all how to use this approach and to show you that it is MORE than possible to accomplish.

In the coming episodes/parts of the series we will covering more general topics such as the one covered here BUT we will be spending a lot of our time learning how to implement this in actually code and learning the skills needed to actually implement these servers in the "real-world".

MDT Exhibit 2038
IPR2025-00542
Page 5

I look forward to going on this journey with you all and seeing all of the incredible things you are able to accomplish along the way.

Onwards and Upwards,

Bradston Henry

==== FOLLOW ME ON SOCIAL MEDIA ====

Twitter: Bradston Dev
Dev.to: @bradstondev
Youtube: Bradston YT
LinkedIn : Bradston Henry

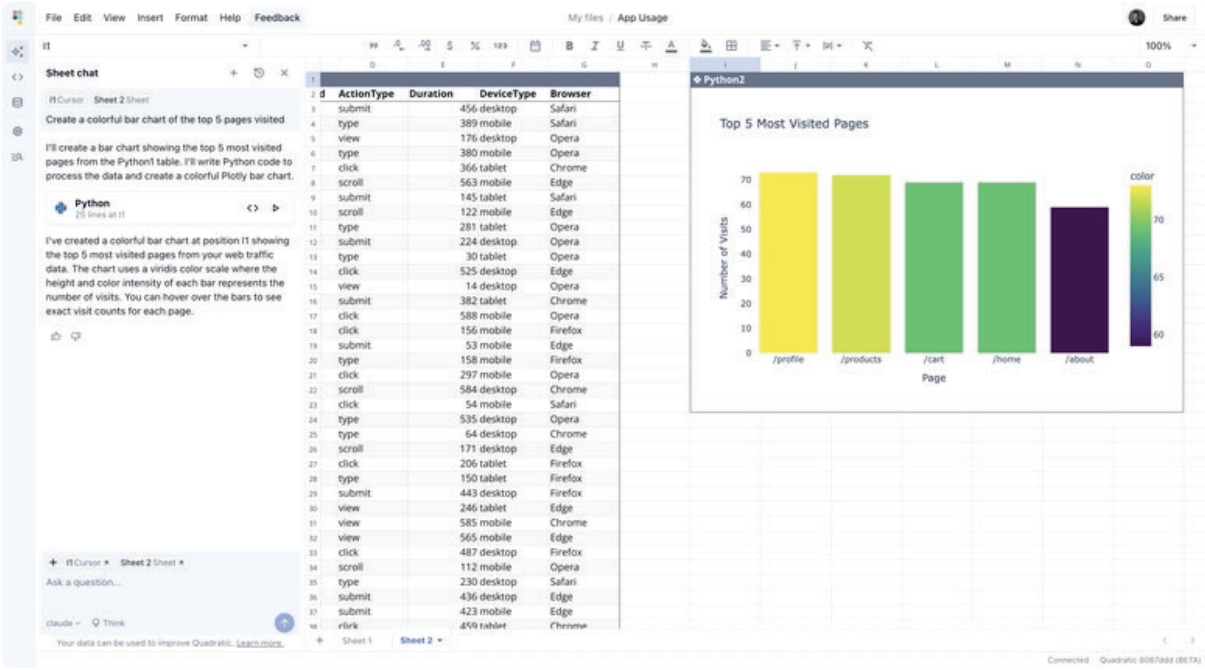
Creating a Multiplayer Game Server (6 Part Series)

1. **Multiplayer Server Basics | Creating a Multiplayer Game Serv...**
2. Persistent vs Non-Persistent Connections | Creating a Multiplayer ...

... 2 more parts...

5. Creating Our Multiplayer Game using Unity | Creating a Multiplay...
6. Deploying Our Multiplayer Game Server to Red Hat OpenShift | C...

Quadratic AI PROMOTED



Free AI chart generator

Upload data, describe your vision, and get Python-powered, AI-generated charts instantly.

Try Quadratic free

Top comments (3)

Konfetti21 • Feb 10 '23

Your article mentions using a local game server. But don't talk about the need for additional costs for such networks. A network with a dedicated server requires one or more more more powerful computers to run dedicated server software. In addition, the server software requires qualified personnel to maintain it.

Luettgen-cpu • Mar 2 '23

MDT Exhibit 2038
IPR2025-00542
Page 7

Comment hidden by post author

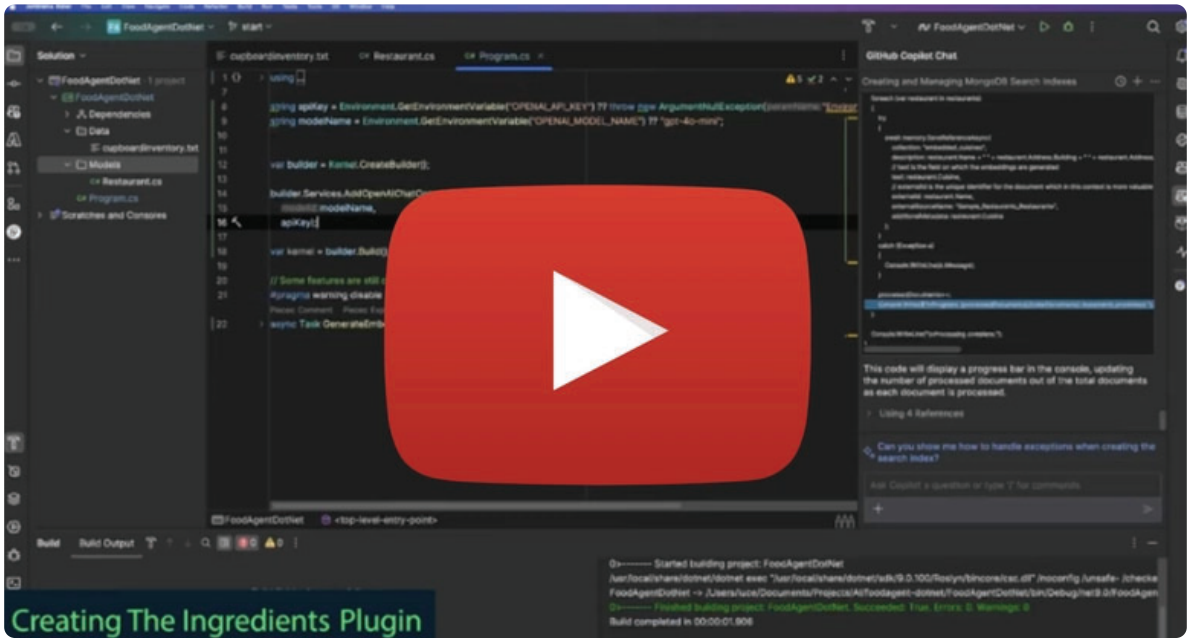
Audrey-cpu • Feb 10 '23

It's not just about this server, the whole difficulty lies in choosing the program itself, which will perform the role of a local server. Suitable options are a huge number, and each of them has its own characteristics. Therefore, it is difficult to say which local server is better than the others.

Some comments have been hidden by the post's author - find out more

Code of Conduct • Report abuse

MongoDB PROMOTED



How to Build an AI Agent with Semantic Kernel (and More!)

Join Developer Advocate Luce Carter for a hands-on tutorial on building an AI-powered dinner recommendation agent. Discover how to integrate Microsoft Semantic Kernel, MongoDB Atlas, C#, and OpenAI for ingredient checks and smart restaurant suggestions.

Watch the video

IBM Developer

More from IBM Developer

What's next in the world of technology? Let's talk about all the hype around distributed cloud!

#cloud #devops #todayilearned #discuss

Deploying Our Multiplayer Game Server to Red Hat OpenShift | Creating a Multiplayer Game Server - Part 6

#gamedev #cloud #unity3d #redhat

Creating Our Multiplayer Game using Unity | Creating a Multiplayer Game Server - Part 5

#gamedev #cloud #unity3d #redhat

Stellar Development Foundation PROMOTED

How a Hackathon Win Led to My Startup Getting Funded

In this episode, you'll see:

- The hackathon wins that sparked the journey.
- The moment José and Joseph decided to go all-in.
- Building a working prototype on Stellar.
- Using the PassKeys feature of Soroban.
- Getting funded via the Stellar Community Fund.

Watch the video 🎥

MDT Exhibit 2038
IPR2025-00542
Page 10