Mod design document

Minecraft plugin

31-03-2025

Dylano, Malik, Mees

Bart Kouwenberg



# Inleiding

Hier in dit document gaan we het hebben over onze gemaakte minecraft plugin. We beginnen met wat ons doel was voor onze plugin. We waren aan het brainstormen en toen kwamen op een spleef gamemode we dachten dat dit leuk was om te maken en te spelen.

# Inhoudsopgave

Inhoudsopgave

[Inleiding 2](#_Toc194312558)

[Inhoudsopgave 3](#_Toc194312559)

[Onze plugin 4](#_Toc194312560)

[Testverslagen 5](#_Toc194312561)

[Logboek 9](#_Toc194312562)

[Nieuwe ontwikkelaren 10](#_Toc194312563)

# Onze plugin

**Uitleg over de Spleef-plugin**

Deze plugin voegt een **Spleef-minigame** toe aan een Minecraft-server. Spleef is een spel waarin spelers proberen de grond onder elkaar weg te graven, zodat tegenstanders in een gat vallen. De laatste speler die overblijft, wint!

**Wat doet de plugin?**

✅ Spelers kunnen zich aanmelden voor Spleef met het commando **/joinspleef**.  
✅ Een speler kan het spel starten met **/startspleef**, als er minstens 2 spelers zijn.  
✅ Spelers worden naar een **Spleef-arena** geteleporteerd.  
✅ Iedere speler krijgt een **betoverde schep** om sneeuwblokken te breken.  
✅ Als een speler valt, wordt die uitgeschakeld.  
✅ De laatste overgebleven speler wint en krijgt een **vuurwerkshow** te zien!

📊 **Scoreboard**:

* Het scoreboard houdt bij wie een ronde heeft gewonnen.

📢 **BossBar voor de wachtrij**:

* Als je je aanmeldt, zie je bovenaan je scherm hoeveel spelers er al in de wachtrij zitten.

**Hoe gebruik je de plugin?**

1️. Gebruik **/joinspleef** om mee te doen.  
2️. Wacht tot een speler **/startspleef** gebruikt om het spel te beginnen.  
3️. Graaf de sneeuw onder je tegenstanders weg en **blijf als laatste over!**  
4️. Geniet van een vuurwerkshow als je wint.

**Samenvatting**

Deze plugin zorgt ervoor dat je **op een leuke manier Spleef kunt spelen** op je server. Spelers kunnen zich aanmelden, het spel starten en proberen als laatste over te blijven. De plugin zorgt automatisch voor de arena, het resetten en het bijhouden van de score! 🎮

# Testverslagen

**Minecraft spleef**

|  |  |  |
| --- | --- | --- |
| **Use Case / Unit**  *Naam van de use case* | Aantal spelers | |
| **Pre-conditie**  *Voorwaarden voor de use case uitgevoerd kan worden* | * De speler joint de server * De speler doe de command /joinspleef * De beheerder start spleef wanneer er minimaal 2 spelers gejoint zijn. | |
| **Beschrijving**  *De flow van de code (happy path)* | De match kan beginnen wanneer er minimaal 2 mensen zijn gejoined in spleef | |
| **Uitzonderingen**  *Wat kan er misgaan?* | Als er maar 1 is gejoined kan je niet startenen dan krijg je de melding van niet geboeg gejoined minimaal 2 mensen moet gejoined zijn omvan start te gaan van spleef | |
| **Post-conditie**  *Wat is de verwachte uitkomst, wat is er gebeurd?* | Dat je de game kan starten met genoeg spelers erin | |
|  | | |
| **Test 1** | **Dataset** | **Resultaat** |
| *Stappen*;   * De speler joint de server * De speler doe de command /joinspleef * De beheerder start spleef wanneer er minimaal 2 spelers gejoint zijn. | *Edge cases(hoog);*  Als je eenmaal bent gejoined, is het niet de bedoeling om opnieuw deel te nemen. Momenteel kan dit wel, waardoor je naam twee keer in het spel wordt opgeslagen. Dit kan verwarring veroorzaken tijdens het spelverloop. | **✔** **❌** |
| *Verwachte uitkomst;*   * De match kan beginnen wanneer er minimaal 2 mensen zijn gejoined in spleef |

|  |  |  |
| --- | --- | --- |
| **Use Case / Unit**  *Naam van de use case* | Spleef spelen | |
| **Pre-conditie**  *Voorwaarden voor de use case uitgevoerd kan worden* | * Wanneer de game start spawn alle spelers op het bovenste platform met een schep * Spelers proberen elkaar aftemaken door de platform onder hun kapot te maken * Als je naar beneden valt ben je uitgeschakeld. * Wie als laatst nog leeft is de winnaar | |
| **Beschrijving**  *De flow van de code (happy path)* | De spelers starten het spel en proberen elkaar af te maken om als laatste te overleven en te gaan winnen | |
| **Uitzonderingen**  *Wat kan er misgaan?* | Speler wordt niet naar het platform getelepoteerd | |
| **Post-conditie**  *Wat is de verwachte uitkomst, wat is er gebeurd?* | De spelers kunnen het spel spleef vlekkeloos spelen | |
|  | | |
| **Test 1** | **Dataset** | **Resultaat** |
| *Stappen*;   * Wanneer de game start spawn alle spelers op het bovenste platform met een schep * Spelers proberen elkaar aftemaken door de platform onder hun kapot te maken * Als je naar beneden valt ben je uitgeschakeld.   Wie als laatst nog leeft is de winnaar | Happy Path;  De spelers starten het spel en proberen elkaar af te maken om als laatste te overleven en te gaan winnen | **✔** **❌** |
| *Verwachte uitkomst;*   * De spelers kunnen het spel spleef vlekkeloos spelen |

|  |  |  |
| --- | --- | --- |
| **Use Case / Unit**  *Naam van de use case* | Scoreboard | |
| **Pre-conditie**  *Voorwaarden voor de use case uitgevoerd kan worden* | * Elke keer wanneer een speler wint komt zijn naam op de scorebord met het getal ernaast van hoe vaak die heeft gewonnen * Wie het meest heeft gewonnen staat boven aan de scoreboard | |
| **Beschrijving**  *De flow van de code (happy path)* | Je kan een overzicht zien wie hoeveel heeft gewonnen en wie boven aan staat met het meeste winst | |
| **Uitzonderingen**  *Wat kan er misgaan?* | Dat speler zijn scoreboard niet wordt bij geteld wanneer die wint | |
| **Post-conditie**  *Wat is de verwachte uitkomst, wat is er gebeurd?* | Iedereen kan de scoreboard zien met hoeveel elke speler heeft gewonnen en wie boven aan staat met het meeste winst | |
|  | | |
| **Test 1** | **Dataset** | **Resultaat** |
| *Stappen*;   * Elke keer wanneer een speler wint komt zijn naam op de scorebord met het getal ernaast van hoe vaak die heeft gewonnen * Wie het meest heeft gewonnen staat boven aan de scoreboard | Edge cases(laag);  Dat speler zijn scoreboard niet wordt bij geteld wanneer die wint | **✔** **❌** |
| *Verwachte uitkomst;*   * Iedereen kan de scoreboard zien met hoeveel elke speler heeft gewonnen en wie boven aan staat met het meeste winst |

# Logboek

Afbeelding met tekst, schermopname, nummer, Lettertype

Door AI gegenereerde inhoud is mogelijk onjuist.

# Nieuwe ontwikkelaren

**Uitleg voor toekomstige ontwikkelaars**

Als je verder wilt werken aan deze Spleef-plugin, is het handig om te weten hoe de code is opgebouwd en waar je verbeteringen of nieuwe functies kunt toevoegen. Hieronder leg ik uit hoe de plugin werkt en waar je dingen kunt aanpassen.

**1. Algemene structuur van de plugin**

De plugin is opgebouwd uit verschillende **belangrijke onderdelen**:

* **onEnable() en onDisable()**: Regelen het opstarten en afsluiten van de plugin.
* **Command-handling**: De plugin reageert op commando’s zoals /joinspleef en /startspleef.
* **Game-logica**: De game start, spelers worden geteleporteerd, krijgen een schep, en de winnaar wordt bepaald.
* **Event-handling**: Bijvoorbeeld wanneer een speler valt, wordt die uitgeschakeld.
* **Arena-management**: De arena wordt gegenereerd en gereset na elke ronde.

De belangrijkste klasse is **Spleef.java**, die bijna alle functionaliteit bevat.

**2. Hoe voeg je een nieuwe functie toe?**

Wil je een nieuwe functie toevoegen? Dan kun je deze stappen volgen:

**Nieuwe commando’s toevoegen**

1. Ga naar de onCommand()-methode in **Spleef.java**.
2. Voeg een nieuwe if-statement toe voor het nieuwe commando.
3. Zorg ervoor dat je logica toevoegt die bepaalt wat het commando doet.
4. Registreer het commando in plugin.yml.

**Voorbeeld:** Stel je wilt een /leavespleef commando maken zodat spelers het spel kunnen verlaten:

if (command.getName().equalsIgnoreCase("leavespleef")) {

if (sender instanceof Player) {

Player player = (Player) sender;

if (spleefPlayers.contains(player)) {

spleefPlayers.remove(player);

player.sendMessage(ChatColor.RED + "Je hebt het Spleef-spel verlaten.");

} else {

player.sendMessage(ChatColor.RED + "Je zit niet in een Spleef-wedstrijd.");

}

}

return true;

}

Dit zorgt ervoor dat een speler zichzelf uit het spel kan halen als die niet meer mee wil doen.

**Nieuwe spelmechanieken toevoegen**

Wil je bijvoorbeeld **power-ups** toevoegen? Dit kan via item drops of speciale blokken.

**Voorbeeld:** Voeg een functie toe waarbij spelers een **speed boost** krijgen als ze een bepaald blok aanraken:

@EventHandler

public void onPlayerStep(PlayerMoveEvent event) {

Player player = event.getPlayer();

if (player.getLocation().getBlock().getType() == Material.GOLD\_BLOCK) {

player.addPotionEffect(new PotionEffect(PotionEffectType.SPEED, 200, 2)); // Speed boost voor 10 seconden

player.sendMessage(ChatColor.GOLD + "Je hebt een speed boost gekregen!");

}

}

Je moet dit dan nog registreren in de onEnable() methode met:

Bukkit.getServer().getPluginManager().registerEvents(this, this);

**3. Hoe verbeter je de bestaande code?**

Wil je de code **efficiënter en netter maken**? Hier zijn een paar suggesties:

✅ **Maak losse klassen voor verschillende onderdelen.**

* **GameManager**: Voor alle game-logica (zoals start/stop en teleporteren).
* **CommandHandler**: Om alle commando’s netjes op te vangen.
* **ArenaManager**: Voor het bouwen en resetten van de arena.

✅ **Gebruik configuratiebestanden (config.yml).**

* Laat de **arena-coördinaten** in een configbestand opslaan in plaats van hardcoded in de code.
* Dit maakt de plugin **makkelijker aanpasbaar** zonder de code te wijzigen.

✅ **Voeg foutafhandeling toe.**

* Controleer of een speler zich al heeft aangemeld voordat ze /joinspleef kunnen gebruiken.
* Geef een duidelijke foutmelding als een commando verkeerd wordt gebruikt.

✅ **Gebruik JavaDocs-commentaar.**

* Voeg uitleg toe aan methodes zodat toekomstige ontwikkelaars snappen wat alles doet.

**4. Waar kun je verder uitbreiden?**

🔹 **Team-Spleef toevoegen** – Laat spelers in teams spelen.  
🔹 **Random evenementen** – Bijvoorbeeld plotseling verdwijnende blokken.  
🔹 **Meer arena's maken** – Voeg een optie toe om uit meerdere arena's te kiezen.  
🔹 **Punten en statistieken bijhouden** – Maak een leaderboard van de beste Spleef-spelers.  
🔹 **GUI-menu’s toevoegen** – Een menu waarin spelers kunnen kiezen of ze meedoen.

**5. Hoe test je de plugin?**

1. **Installeer een testserver met Paper.**
2. **Zet de plugin (.jar bestand) in de plugins-map.**
3. **Start de server en kijk of er foutmeldingen in de console zijn.**
4. **Test alle commando’s in Minecraft.**
5. **Controleer of de game goed start, eindigt en reset.**
6. **Verbeter de code als er bugs zijn!**

**Samenvatting**

🔹 De code is goed opgebouwd, maar kan verbeterd worden door **losse klassen** en **config-bestanden** te gebruiken.  
🔹 Nieuwe functies zoals **power-ups, team-Spleef of extra arena’s** kunnen makkelijk worden toegevoegd.  
🔹 **Test de plugin goed** voordat je het op een echte server gebruikt.  
🔹 Houd de code **netjes en overzichtelijk**, zodat anderen het makkelijk kunnen aanpassen.

Veel succes met ontwikkelen! 🚀