

# Boom Online Crazy Arcade



Bạn có biết game Boom Online đặt bom sắc nước hay còn gọi là Crazy Arcade?



Có, mình biết game **Boom Online** (còn gọi là **Crazy Arcade**). Đây là một trò chơi đặt bom nước rất nổi tiếng do công ty **Nexon** (Hàn Quốc) phát triển, từng rất phổ biến ở Việt Nam vào khoảng đầu những năm 2000.

## Một số điểm nổi bật của Boom Online:

- **Lối chơi:** Tương tự Bomberman, nhưng thay vì bom nổ, người chơi đặt **bóng nước** (bong bóng) để "nhốt" đối thủ. Khi bị dính nước, nhân vật sẽ bị nhốt trong quả bong bóng và bay lơ lửng trong vài giây. Đồng đội có thể cứu hoặc kẻ địch có thể "kết liễu".
- **Nhân vật:** Nhiều nhân vật đáng yêu như Dao, Keo, Béo, Bố Đầu Nhỏ... với kỹ năng và diện mạo riêng.
- **Đồ họa:** Phong cách **chibi**, dễ thương, đầy màu sắc.
- **Chế độ chơi:** Có nhiều chế độ chơi như đặt bom truyền thống, sinh tồn, đối kháng đội, chơi cùng AI...
- **Trang bị và vật phẩm:** Nhiều loại vật phẩm hỗ trợ như giày chạy nhanh, tăng sức nước, tăng số lượng bóng...

## Game này ảnh hưởng như thế nào?

Boom Online từng là một phần tuổi thơ của nhiều game thủ Việt Nam. Ngoài giải trí, nó còn gắn liền với các **phòng net**, **các buổi chơi cùng bạn bè**, tạo ra rất nhiều kỷ niệm đáng nhớ.

---

Bạn đang muốn tìm lại game này để chơi, hay muốn **làm một game tương tự bằng Unity?**

Tuyệt vời, Defold là một game engine nhẹ, mạnh mẽ, và rất phù hợp để làm các game 2D như Boom Online. Mình sẽ hướng dẫn bạn làm một phiên bản đơn giản trước — ví dụ như 1 người chơi, có thể di chuyển và đặt bom nước, bom nổ lan ra theo các hướng, làm vỡ gạch, và có thể mở rộng ra chơi nhiều người sau.

## Tổng quan các bước tạo game kiểu Boom Online bằng Defold:

---

### Bước 1: Chuẩn bị môi trường

- Cài [Defold](#)
- Tạo một **project mới**
- Cấu trúc thư mục nên có:

```
/main
- main.collection
- player.script
- bomb.script
- explosion.script
- tilemap.tilesources
- map.tilemap
```

---

### Bước 2: Tạo bản đồ (Tilemap)

- Dùng **Tilemap** với các ô vuông kích thước như 16x16 hoặc 32x32.

- Vẽ các khối:
    - Gạch cứng (không phá được)
    - Gạch mềm (phá được khi nổ)
    - Đất trống (có thể đi và đặt bom)
- 

### Bước 3: Tạo nhân vật

- Tạo một Game Object có:
    - ``sprite`` nhân vật
    - ``collision object`` dạng box
  - Viết ``player.script`` xử lý:
    - Di chuyển (bằng phím mũi tên hoặc WASD)
    - Đặt bom (khi nhấn Space)
- 

### Bước 4: Bom nước

- Khi đặt bom:
  - Spawn 1 Game Object ``bomb.go``
  - Sau khoảng 1-2 giây, bom nổ
- ``bomb.script`` xử lý:
  - Gửi tín hiệu nổ (spawn các "vết nước" theo 4 hướng)
  - Phá gạch mềm nếu gặp
  - Nếu gặp tường cứng thì dừng lan

- Nếu chạm vào người chơi thì bị “úp nước”
- 

### Bước 5: Hiệu ứng nước nổ (Explosion)

- ``explosion.script`` tạo hiệu ứng lan ra 4 hướng
  - Có thể dùng animation sprite (bóng nước) để hiển thị
  - Mỗi vết nước tồn tại trong 0.3–0.5 giây rồi biến mất
- 

### Bước 6: Mở rộng

- Thêm item (giày, tăng bom, tăng nước)
  - Thêm nhiều người chơi
  - Thêm AI
  - Thêm hiệu ứng âm thanh, âm nhạc
- 

Mình có thể bắt đầu giúp bạn tạo từng phần. Bạn muốn bắt đầu với bản đồ, nhân vật, hay đặt bom trước?



làm chi tiết từng phần, bắt đầu là bản đồ, rồi tới nhân vật di chuyển, và đặt bom bằng space



Rất hay, chúng ta sẽ đi **từng bước một cách chi tiết**, bắt đầu từ **bản đồ (Tilemap)** trong Defold. Mình sẽ giải thích rõ từng phần để bạn hiểu sâu và tự chỉnh sửa được sau này.

---

## **BƯỚC 1: TẠO BẢN ĐỒ (TILEMAP)**

### **Mục tiêu:**

Tạo một bản đồ đơn giản với các ô:

- **Tường cứng** (không phá được)
  - **Tường mềm** (phá được khi bom nổ)
  - **Đất trống** (có thể đi vào)
- 

### **Các bước thực hiện:**

#### **1. Tạo Project Mới:**

- Mở Defold → ``New Project``
- Đặt tên như ``BoomClone``
- Mở ``main/main.collection`` (đây là "cảnh chính" của game)

#### **2. Tạo Tileset:**

- Chuột phải vào thư mục ``main`` → ``New`` → ``Tile Source``
- Đặt tên: ``tiles.tilesource``

**Chọn hình ảnh tileset:**

- Dùng hình 16x16 hoặc 32x32 gồm 3 loại gạch (tường cứng, tường mềm, đất trống).
- Nếu chưa có, bạn có thể dùng tạm hình test như:
  - Tường cứng: màu xám
  - Tường mềm: nâu
  - Đất trống: xanh nhạt

**Trong ``tiles.tilesource``:**

- Kéo hình ảnh vào
- Đặt:
  - Tile Width: ``32``
  - Tile Height: ``32``
  - Margin / Spacing: ``0``

Lưu lại (``ctrl+s``)

### 3. Tạo Tilemap:

- Chuột phải ``main`` → ``New`` → ``Tilemap``
- Đặt tên: ``map.tilemap``
- Trong ``map.tilemap``:
  - ``Tile Set``: chọn ``tiles.tilesource``
  - Layer: để mặc định là ``layer1``
  - Width/Height: ví dụ ``15 x 13``
  - Tile Size: ``32 x 32``

### Vẽ bản đồ:

- Dùng công cụ vẽ để tạo một bản đồ:
  - Tường cứng bao quanh
  - Xen kẽ tường mềm
  - Chừa các ô đất trống để đi

#### 4. Thêm Tilemap vào Scene:

- Trong ``main.collection``, chuột phải → ``Add Game Object File``
- Tạo mới Game Object → đặt tên ``map.go``
- Trong ``map.go``:
  - Add Component → ``Tilemap`` → chọn ``map.tilemap``
- Kéo ``map.go`` vào ``main.collection``

#### 5. Thêm Collision (cho sau):

- Sau khi có người chơi và bom, ta sẽ dùng tilemap này để kiểm tra va chạm (bằng script, không cần collider cho từng ô)

---

### ✅ Xong phần tạo bản đồ!

Bạn đã sẵn sàng để sang phần **nhân vật di chuyển** chưa? Mình sẽ hướng dẫn tạo player và điều khiển bằng bàn phím.



ok nhân vật di chuyển đi



Tuyệt, giờ chúng ta sẽ làm phần **nhân vật có thể di chuyển** trong bản đồ.



## BƯỚC 2: TẠO NHÂN VẬT DI CHUYỂN



### Mục tiêu:

- Tạo một nhân vật có thể di chuyển bằng **phím mũi tên hoặc WASD**
- Nhân vật sẽ không đi xuyên tường (ta sẽ xử lý sau bằng tilemap check)



### Các bước thực hiện:

#### 1. Tạo Sprite Nhân Vật:

Bạn cần một hình ảnh nhân vật (ví dụ PNG 32x32). Có thể là hình vuông tạm thời để test.

- Chuột phải thư mục ``main`` → ``New`` → ``Atlas`` → đặt tên: ``player.atlas``
- Kéo hình ảnh nhân vật vào atlas

#### 2. Tạo Game Object Nhân Vật:

- Chuột phải ``main`` → ``New`` → ``Game Object`` → đặt tên: ``player.go``
- Trong ``player.go``, thêm các component:

##### 1. Sprite:

- Image: chọn ``player.atlas``
- Default Animation: tên ảnh trong atlas





- Material: ``builtins/sprite.material``
- Blend Mode: ``Alpha``

## 2. **Collision Object** (kiểu Kinematic):

- Type: **Kinematic**
  - Shape: Box (Size: 32x32)
  - Group: ``player``
  - Mask: ``wall`` (ta sẽ thêm tường sau)
- Add script: ``player.script`` (mình sẽ tạo tiếp dưới đây)

## 3. Script Di Chuyển - ``player.script``

Tạo script: chuột phải ``main`` → ``New`` → ``Script`` → đặt tên ``player.script``, rồi dán nội dung sau:

```
go.property("speed", 100)

function init(self)
    msg.post(".", "acquire_input_focus")
    self.dir = vmath.vector3()
end

function on_input(self, action_id, action)
    self.dir = vmath.vector3()
    if action_id == hash("up") and action.pressed or action.repeated then
        self.dir.y = 1
    elseif action_id == hash("down") and action.pressed or action.repeated then
        self.dir.y = -1
    elseif action_id == hash("left") and action.pressed or action.repeated then
        self.dir.x = -1
    elseif action_id == hash("right") and action.pressed or action.repeated
then
```

```
        self.dir.x = 1
    end
end

function update(self, dt)
    local movement = self.dir * self.speed * dt
    go.set_position(go.get_position() + movement)
end
```

---

## 4. Cấu hình Input:

Mở `game.input_binding` → Thêm các phím:

- `up` → `Up Arrow` và `W`
- `down` → `Down Arrow` và `S`
- `left` → `Left Arrow` và `A`
- `right` → `Right Arrow` và `D`

---

## 5. Thêm Player vào Scene:

- Mở `main.collection`
- Kéo `player.go` vào
- Đặt vị trí phù hợp (ví dụ ô đất trống)

✅ **Kết quả:** Bạn có thể chạy game (``ctrl+B``) và điều khiển nhân vật bằng phím mũi tên/WASD!

---

Bước kế tiếp là **đặt bom bằng phím Space**, bạn muốn tiếp tục chứ?



ok



Tuyệt! Bây giờ ta sẽ làm phần **đặt bom bằng phím Space**, bom sau vài giây sẽ **nổ và xóa chính nó**. Đây là nền tảng quan trọng để sau này thêm hiệu ứng nước lan, phá gạch, v.v.

---

## 💣 BƯỚC 3: ĐẶT BOM BẰNG PHÍM SPACE

### 🎯 Mục tiêu:

- Nhấn **phím Space** thì nhân vật **đặt bom** ở ô hiện tại.
  - Sau 2 giây bom sẽ **nổ và biến mất** (hiện tại chỉ test đơn giản).
- 

### 🔧 Các bước thực hiện:

---

#### 1. Tạo Sprite Bom:

- Chuột phải ``main`` → ``New`` → ``Atlas`` → đặt tên: ``bomb.atlas``
  - Kéo hình bom (hoặc ảnh tạm 32x32 màu đen) vào
- 

## 2. Tạo Game Object cho Bom:

- Chuột phải ``main`` → ``New`` → ``Game Object`` → tên: ``bomb.go``
  - Thêm các component:
    1. **Sprite:**
      - Image: ``bomb.atlas``
      - Default animation: chọn ảnh
    2. **Script:** tạo script mới tên ``bomb.script`` (nội dung bên dưới)
- 

## 3. Script bom - ``bomb.script``:

```
function init(self)
  -- Sau 2 giây thì nổ (ở đây là xóa bom)
  timer.delay(2.0, false, function()
    go.delete()
    -- sau này sẽ spawn vụ nổ ở đây
  end)
end
```

## 4. Cho Player đặt bom:

Mở lại ``player.script``, sửa như sau:

### Thêm ở đầu:

```
local bomb_factory = "#bomb_factory"
```

### Trong `init(self)` thêm:

```
self.can_place_bomb = true
```

### Trong `on\_input`, thêm:

```
if action_id == hash("place_bomb") and action.pressed and self.can_place_bomb
then
  -- Lấy vị trí player, làm tròn theo tile
  local pos = go.get_position()
  pos.x = math.floor(pos.x / 32 + 0.5) * 32
  pos.y = math.floor(pos.y / 32 + 0.5) * 32

  factory.create(bomb_factory, pos)

  self.can_place_bomb = false
  -- Cho phép đặt tiếp sau 2 giây
  timer.delay(2.0, false, function()
    self.can_place_bomb = true
  end)
end
```

## 5. Thêm Bom Factory vào `player.go`:

- Chuột phải `player.go` → `Add Component` → **Factory**
- `Prototype`: chọn `bomb.go`
- Đặt ID: `bomb\_factory`

---

## 6. Thêm Input Space:

- Mở `game.input_binding``
- Add action:
  - Action: `place_bomb``
  - Key Trigger: `Space``

---

✅ **Chạy game:** Nhân vật có thể di chuyển và đặt bom bằng phím Space. Bom xuất hiện đúng ô vuông và biến mất sau 2 giây!

---

Bước tiếp theo là **bom nổ tạo hiệu ứng nước lan 4 hướng và phá gạch mềm**. Bạn muốn tiếp tục luôn không?



ok



Tuyệt vời! Giờ ta sẽ làm phần **bom nổ tạo hiệu ứng nước lan ra 4 hướng** (trên, dưới, trái, phải) – giống phong cách Boom Online.

## 💣 BƯỚC 4: HIỆU ỨNG NƯỚC NỔ LAN RA 4 HƯỚNG

### 🎯 Mục tiêu:

- Khi bom nổ, tạo **hiệu ứng nước** (gọi là "explosion") lan ra theo 4 hướng.
- Nước sẽ dừng lại nếu gặp **tường cứng**.
- Gạch mềm sẽ **bị phá** (chúng ta sẽ thêm ở bước sau).

### 🔧 Các bước thực hiện:

#### 1. Tạo Sprite Hiệu Ứng Nước Nổ:

- Chuột phải `main` → `New` → `Atlas` → tên: `explosion.atlas`
- Kéo hình ảnh hiệu ứng nước vào (hoặc dùng hình tạm màu xanh)
- Gồm:
  - Hình giữa (center)
  - Đầu tia nước (up, down, left, right)
  - Thân tia nước (mid-left, mid-right, mid-up, mid-down) – có thể đơn giản hóa thành 1 loại

#### 2. Tạo Explosion GO:

- Chuột phải `main` → `New` → `Game Object` → tên: `explosion.go`
- Thêm `Sprite` → dùng `explosion.atlas`

- Thêm ``Script``: tên ``explosion.script`` (nội dung bên dưới)
- 

### 3. Script explosion - ``explosion.script``:

```
function init(self)
  -- Tự hủy sau 0.4 giây
  timer.delay(0.4, false, function()
    go.delete()
  end)
end
```

### 4. Tạo Explosion Factory:

- Chuột phải ``main`` → ``New`` → ``Factory`` → tên ``explosion.factory``
  - Prototype: ``explosion.go``
- 

### 5. Gắn explosion factory vào ``bomb.go``:

- Chuột phải ``bomb.go`` → ``Add Component`` → chọn ``Factory``
  - Prototype: ``explosion.go``
  - ID: ``explosion_factory``
- 

### 6. Sửa ``bomb.script`` để spawn nước lan ra:



Thay `init(self)` bằng:

```
local TILE_SIZE = 32
local RANGE = 2 -- tâm lan bom

function explode(pos)
    -- Hiệu ứng giữa
    factory.create("#explosion_factory", pos)

    -- 4 hướng
    local dirs = {
        vmath.vector3(1, 0, 0), -- phải
        vmath.vector3(-1, 0, 0), -- trái
        vmath.vector3(0, 1, 0), -- lên
        vmath.vector3(0, -1, 0), -- xuống
    }

    for _, dir in ipairs(dirs) do
        for i = 1, RANGE do
            local p = pos + dir * TILE_SIZE * i
            -- TODO: sau này check map để dừng nếu gặp tường
            factory.create("#explosion_factory", p)
        end
    end
end

function init(self)
    timer.delay(2.0, false, function()
        local pos = go.get_position()
        explode(pos)
        go.delete()
    end)
end
```

✅ **Chạy game:** Bom nổ sau 2 giây, tạo hiệu ứng lan 4 hướng (như nước Boom Online).

Hiện tại nước sẽ lan xuyên mọi thứ. Bước tiếp theo ta sẽ:

1. **Dừng nước nếu gặp tường cứng**
2. **Phá gạch mềm**
3. (Sau đó) **Hạ gục người chơi nếu dính nước**

---

Bạn muốn mình hướng dẫn phần **tường cứng, tường mềm, và chặn nước lại khi gặp tường** tiếp luôn không?



ok



Tốt, chúng ta sẽ xử lý **va chạm nước với bản đồ**: nếu nước gặp **tường cứng** thì **dừng lại**, nếu gặp **tường mềm** thì **phá được** và **dừng lại** tại đó (giống Boom Online).

---

## **BƯỚC 5: CHẶN NƯỚC BỞI TƯỜNG CỨNG VÀ PHÁ GẠCH MỀM**

### **Mục tiêu:**

- Xác định **tường cứng không thể phá** và **gạch mềm có thể phá** từ Tilemap.
- Nước nổ lan đến đâu thì kiểm tra tile:
  - Nếu là **tường cứng** → dừng ngay, không tạo explosion.

- Nếu là **gạch mềm** → tạo explosion và phá tile đó.
- Nếu là đất trống → tạo explosion và tiếp tục lan.

---


## Các bước thực hiện:

---

### 1. Chuẩn hóa Tilemap để nhận dạng các loại gạch

Giả sử trong ``tiles.tilesource``, bạn có các tile ID như sau:

- Tile ``0``: đất trống
- Tile ``1``: tường cứng
- Tile ``2``: tường mềm (phá được)

 **Bạn có thể xem Tile ID** bằng cách mở ``map.tilemap``, nhấn vào từng ô trong layer — ID sẽ hiển thị ở góc.

---

### 2. Thêm hàm kiểm tra tile trong ``bomb.script``:

Ta cần biết tile tại 1 vị trí là loại nào. Thêm đoạn sau vào ``bomb.script``:

```
local TILE_SIZE = 32
local RANGE = 2
local TILEMAP_URL = "main:/map" -- Đường dẫn đến tilemap component

local TILE_EMPTY = 0
local TILE_HARD = 1
local TILE_SOFT = 2

-- Trả về tile ID tại vị trí world
```

```

local function get_tile_at(pos)
    local tile_x = math.floor(pos.x / TILE_SIZE) + 1
    local tile_y = math.floor(pos.y / TILE_SIZE) + 1
    return tilemap.get_tile(TILEMAP_URL, "layer1", tile_x, tile_y), tile_x,
tile_y
end

-- Xóa tile mềm (set về tile trống)
local function destroy_tile(x, y)
    tilemap.set_tile(TILEMAP_URL, "layer1", x, y, TILE_EMPTY)
end

```

### 3. Sửa hàm `explode()` để kiểm tra từng ô tile:

Thay hàm `explode()` bằng đoạn sau:

```

function explode(pos)
    -- Hiệu ứng giữa
    factory.create("#explosion_factory", pos)

    local dirs = {
        vmath.vector3(1, 0, 0), -- phải
        vmath.vector3(-1, 0, 0), -- trái
        vmath.vector3(0, 1, 0), -- lên
        vmath.vector3(0, -1, 0), -- xuống
    }

    for _, dir in ipairs(dirs) do
        for i = 1, RANGE do
            local p = pos + dir * TILE_SIZE * i
            local tile, tx, ty = get_tile_at(p)

            if tile == TILE_HARD then
                break -- Gặp tường cứng → dừng
            elseif tile == TILE_SOFT then
                destroy_tile(tx, ty)
            end
        end
    end
end

```

```
        factory.create("#explosion_factory", p)
        break -- Gặp gạch mềm → phá và dừng
    else
        factory.create("#explosion_factory", p)
    end
end
end
end
```

### ✅ Chạy thử:

- Nếu đặt bom gần tường cứng → nước không đi qua.
- Nếu gần gạch mềm → gạch bị phá sau khi bom nổ.

### 🎯 Tuyệt! Giờ bom của bạn đã:

- Nổ đúng vị trí
- Tạo nước lan
- Phá gạch mềm và dừng ở đó
- Không xuyên tường

Bạn muốn mình hướng dẫn **nước làm nhân vật “chết” (bay lên bong bóng)** như Boom Online tiếp không?





Tuyệt! Giờ ta sẽ thêm cơ chế **nhân vật bị tiêu diệt nếu đứng trong vùng nước nổ** – giống Boom Online khi bị trúng bom, sẽ bị **“bắt” vào bong bóng**.

## □ BƯỚC 6: NHÂN VẬT DÍNH NƯỚC BỊ “CHẾT” (VÀO BONG BÓNG)

### 🎯 Mục tiêu:

- Khi **explosion xuất hiện**, kiểm tra xem có **va chạm với player** không.
- Nếu có → player bị “hạ” (bị đưa vào bong bóng hoặc biến mất).
- Có thể thay bằng hiệu ứng hoặc làm game over đơn giản.

### 🔧 Cách thực hiện:

#### 1. Thêm Collision cho `explosion.go`:

Mở `explosion.go`, thêm component:

- **Collision Object**
  - Type: **Trigger**
  - Shape: Box (size 32x32)
  - Group: `explosion`

- Mask: ``player`` (để va chạm với player)
- 

## 2. Trong ``player.go``:

Đảm bảo player có ``Collision Object``:

- Type: **Kinematic**
  - Group: ``player``
  - Mask: ``explosion`` (để nhận va chạm)
- 

## 3. Bắt va chạm trong ``player.script``:

Thêm vào script của player:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("trigger_response") then
        for _, other in ipairs(message.enter) do
            if other.group == hash("explosion") then
                die(self)
            end
        end
    end
end

function die(self)
    -- Tắt input, xóa nhân vật, hoặc hiện hiệu ứng bong bóng
    print("Player died!")
    msg.post(".", "release_input_focus")
    go.delete()
```

```
-- Hoặc: bạn có thể hiện bong bóng tại vị trí player ở đây  
end
```

## 4. (Tuỳ chọn) Hiện hiệu ứng bong bóng:

Nếu bạn có ảnh "bong bóng", có thể:

- Tạo ``bubble.go`` với sprite bong bóng
- Dùng ``factory.create()`` tại vị trí nhân vật trong ``die(self)``

Ví dụ:

```
local pos = go.get_position()  
factory.create("#bubble_factory", pos)
```

## ✅ Chạy game thử:

- Di chuyển nhân vật vào vùng nước nổ.
- Nếu đúng, bạn sẽ thấy dòng ``"Player died!"`` hoặc nhân vật biến mất.

🎉 **Xong rồi!** Giờ bạn đã có:

- Nhân vật di chuyển
- Bom nổ đúng cơ chế
- Nước lan và chặn bởi tường



- Nhân vật bị tiêu diệt khi dính nước

---

Bạn muốn mình tổng kết lại toàn bộ hệ thống hoặc tiếp tục hướng dẫn phần gì tiếp theo (ví dụ: nhiều người chơi, AI bot, item, giao diện...)?



ai bot



Tuyệt, giờ ta sẽ làm phần **AI bot biết di chuyển và đặt bom phá gạch** — giống như một người chơi máy trong Boom Online.



## BƯỚC 7: THÊM BOT AI CƠ BẢN



### Mục tiêu:

- Tạo **nhân vật bot** có thể:
  - Tự đi **lang thang**
  - Khi thấy **gạch mềm gần**, sẽ đặt bom để phá
- Chưa cần thông minh tránh bom — ta làm từng bước



### Các bước thực hiện:



## 1. Tạo `bot.go` (game object bot):

- Copy lại `player.go` → đổi tên thành `bot.go`
  - Giữ lại:
    - Sprite
    - Collision
    - Factory đặt bom
  - Đổi script thành `bot.script` (sẽ tạo sau)
- 

## 2. Tạo `bot.script`:

Chuột phải → `New Script` → tên: `bot.script`

Dán nội dung sau:

```
go.property("speed", 60)

local TILE_SIZE = 32
local TILEMAP_URL = "main:/map"
local TILE_SOFT = 2
local TILE_EMPTY = 0

local MOVE_DELAY = 0.4
local BOMB_DELAY = 1.5

function init(self)
    self.dir = vmath.vector3()
    self.timer = 0
    self.bomb_timer = 0
end

-- kiểm tra tile tại hướng
local function get_tile_at_dir(pos, dir)
```

```

    local check_pos = pos + dir * TILE_SIZE
    local tile_x = math.floor(check_pos.x / TILE_SIZE) + 1
    local tile_y = math.floor(check_pos.y / TILE_SIZE) + 1
    return tilemap.get_tile(TILEMAP_URL, "layer1", tile_x, tile_y), check_pos,
tile_x, tile_y
end

function update(self, dt)
    self.timer = self.timer - dt
    self.bomb_timer = self.bomb_timer - dt

    local pos = go.get_position()

    if self.timer <= 0 then
        -- Chọn hướng ngẫu nhiên: trái/phải/lên/xuống
        local dirs = {
            vmath.vector3(1, 0, 0),
            vmath.vector3(-1, 0, 0),
            vmath.vector3(0, 1, 0),
            vmath.vector3(0, -1, 0)
        }

        for _, dir in ipairs(dirs) do
            local tile, _, tx, ty = get_tile_at_dir(pos, dir)
            if tile == TILE_SOFT and self.bomb_timer <= 0 then
                -- Gấp gạch mềm → đặt bom
                local p = vmath.vector3(
                    math.floor(pos.x / TILE_SIZE + 0.5) * TILE_SIZE,
                    math.floor(pos.y / TILE_SIZE + 0.5) * TILE_SIZE,
                    0
                )
                factory.create("#bomb_factory", p)
                self.bomb_timer = BOMB_DELAY
                return
            end
        end
    end

    -- Nếu không thấy gạch mềm, di chuyển ngẫu nhiên
    local dir = dirs[math.random(1, #dirs)]
    local tile, target_pos = get_tile_at_dir(pos, dir)

```

```

        if tile == TILE_EMPTY then
            self.dir = dir
            self.timer = MOVE_DELAY
        else
            self.dir = vmath.vector3() -- đứng yên
        end
    end

    -- Di chuyển
    local movement = self.dir * self.speed * dt
    go.set_position(pos + movement)
end

```

### 3. Thêm `bot.go` vào `main.collection`:

- Kéo vào map, đặt vào vùng đất trống

### 4. Test Game:

- Chạy game (`Ctrl+B`)
- Bot sẽ:
  - Tự đi qua lại
  - Nếu gặp gạch mềm → đặt bom để phá

### Nâng cấp sau:

- Tránh nước nổ (hiện tại chưa né bom)

- Tìm đường thông minh
- Tấn công người chơi

---

Bạn muốn mình làm tiếp phần **AI tránh bom và nước nổ**, hay **thêm nhiều bot cùng lúc**, hay sang phần khác như **item rơi ra từ gạch mềm**?



ai tránh bom



Rất hay! Ta sẽ làm **AI biết tránh bom** – một bước quan trọng giúp bot thông minh hơn, không tự sát như trước.

---

## BƯỚC 8: AI TRÁNH BOM VÀ NƯỚC NỔ

### Mục tiêu:

- Khi bot sắp dính **bom nổ**, nó sẽ:
  - **Tìm hướng an toàn** để chạy
  - **Tạm ngừng đặt bom** khi đang chạy trốn

---

### Cách hoạt động:

1. Mỗi frame, AI sẽ **quét vùng nước sắp nổ** (dựa vào vị trí các quả bom).

2. Nếu vị trí bot đang đứng hoặc sắp đứng bị nguy hiểm, thì:

- Không đặt bom
- Chạy về hướng an toàn

---


## Thực hiện từng bước:

---

### 1. Bot nhận thông tin từ bom (khi tạo):

Mở `bomb.script`, trong `init(self)`:

```
-- Gửi thông tin vị trí và thời điểm nổ cho bot
msg.post("main:/bot#bot", "bomb_created", {
    x = math.floor(go.get_position().x / 32 + 0.5),
    y = math.floor(go.get_position().y / 32 + 0.5),
    time = socket.gettime() + 2 -- nổ sau 2 giây
})
```

 (Bạn cần thêm `local socket = require("socket")` ở đầu script)

---

### 2. Bot lưu danh sách bom hiện có:

Trong `bot.script`, thêm ở đầu:

```
local socket = require("socket")
self.bombs = {}
```

Trong `on_message(self, message_id, message, sender)`:

```

if message_id == hash("bomb_created") then
    table.insert(self.bombs, {
        x = message.x,
        y = message.y,
        explode_time = message.time
    })
end

```

### 3. Thêm hàm kiểm tra ô tile có bị nổ không:

Thêm hàm sau vào `bot.script`:

```

local function is_tile_dangerous(x, y, current_time)
    for _, bomb in ipairs(self.bombs) do
        if bomb.explode_time - current_time <= 0.5 then
            if bomb.x == x and bomb.y == y then
                return true
            end
            -- Check 4 hướng nước lan
            for dx = -2, 2 do
                if dx ~= 0 and bomb.x + dx == x and bomb.y == y then return
true end
            end
            for dy = -2, 2 do
                if dy ~= 0 and bomb.x == x and bomb.y + dy == y then return
true end
            end
        end
    end
    return false
end

```

#### 4. Thay thế đoạn chọn hướng trong `update(self, dt)` bằng:

```
local time_now = socket.gettime()

-- Vị trí hiện tại
local tile_x = math.floor(pos.x / TILE_SIZE + 0.5)
local tile_y = math.floor(pos.y / TILE_SIZE + 0.5)

if is_tile_dangerous(tile_x, tile_y, time_now) then
    -- Bot đang trong vùng nguy hiểm → tìm đường thoát
    local dirs = {
        vmath.vector3(1, 0, 0),
        vmath.vector3(-1, 0, 0),
        vmath.vector3(0, 1, 0),
        vmath.vector3(0, -1, 0)
    }

    for _, dir in ipairs(dirs) do
        local tx = tile_x + dir.x
        local ty = tile_y + dir.y
        local tile = tilemap.get_tile(TILEMAP_URL, "layer1", tx, ty)

        if tile == TILE_EMPTY and not is_tile_dangerous(tx, ty, time_now) then
            -- Chạy về hướng an toàn
            self.dir = dir
            self.timer = MOVE_DELAY
            return
        end
    end

    -- Nếu không có chỗ nào an toàn → đứng yên chịu chết 🤖
    self.dir = vmath.vector3()
    return
end
```

✅ Chạy thử:



- Bot sẽ **không đứng trong vùng nước sắp nổ**
  - **Chạy trốn bom** nếu bị đe dọa
  - Không đặt bom khi đang trốn
- 

🎉 **Kết quả:** Bot giờ đã có logic "sinh tồn" cơ bản.

---

Bạn muốn tiếp tục với:

- **Bot tìm và đuổi theo người chơi**
- **Thêm nhiều bot & đấu với nhau**
- **Item rơi từ gạch mềm**
- **Giao diện thắng/thua?**

Bạn muốn hướng tiếp theo là gì?



làm item đi



Tuyệt! Giờ ta sẽ làm hệ thống **Item rơi ra từ gạch mềm** sau khi bị phá — giống như trong Boom Online.

---

## BƯỚC 9: THÊM ITEM RƠI RA TỪ GẠCH MỀM

### Mục tiêu:

- Khi **gạch mềm bị phá**, có **xác suất rơi item** (tăng bom, tăng nước, tăng tốc, v.v.)
  - Người chơi đi vào **ăn item** thì được **buff**
- 

### Các bước thực hiện:

---

#### 1. Tạo item Game Object (``item.go``)

##### Bao gồm:

- 1 ``Sprite`` (chọn hình item, ví dụ: tăng bom)
  - 1 ``Collision Object``:
    - Type: **Trigger**
    - Group: ``item``
    - Mask: ``player``, ``bot``
- 

#### 2. Tạo Factory để sinh item

Trong một ``game object`` như ``main``, thêm:

- ``Factory`` → tên: ``item_factory``
- Prototype: chọn ``item.go``

### 3. Sửa hàm phá gạch mềm trong `bomb.script`


Ở hàm `explode()` trong đoạn:

```
destroy_tile(tx, ty)
factory.create("#explosion_factory", p)
```

→ Đổi thành:

```
destroy_tile(tx, ty)
factory.create("#explosion_factory", p)

-- Tỷ lệ rơi item (ví dụ: 30%)
if math.random() < 0.3 then
    factory.create("main#item_factory", p)
end
```

 Bạn cần thêm `math.randomseed(os.time())` trong `init()` để đảm bảo random khác nhau.

### 4. Script cho `item.go` (item.script):

```
function on_message(self, message_id, message, sender)
    if message_id == hash("trigger_response") then
        for _, other in ipairs(message.enter) do
            if other.group == hash("player") or other.group == hash("bot") then
                -- Gửi thông điệp đến nhân vật nhận item
                msg.post(other.id, "item_collected", { type = "powerup" })
                go.delete()
            end
        end
    end
end
```

```
end  
end
```

## 5. Nhận item trong player và bot

Trong `player.script` và `bot.script`, thêm:

```
function on_message(self, message_id, message, sender)  
    if message_id == hash("item_collected") then  
        print("Got item:", message.type)  
        -- Tùy loại item có thể xử lý khác nhau  
        self.speed = self.speed + 20  
    end  
end
```

### ✅ Test Game:

- Đặt bom phá gạch mềm
- Khoảng 30% sẽ rơi ra item
- Nếu nhân vật đi vào item → item biến mất + tăng tốc

### 🧠 Có thể mở rộng:

- Nhiều loại item khác nhau:
  - `"bomb\_up"`: tăng số lượng bom

- `"range_up"`: tăng độ lan của nước
- `"speed_up"`: tăng tốc độ
- Thêm animation, xoay item, âm thanh khi ăn

---

Bạn muốn mình hướng dẫn làm **nhiều loại item** hay **item có hiệu ứng xoay & phát sáng**, hay đi tiếp phần **thắng/thua** và **hiện UI**?



làm 3 loại item cơ bản thôi: số bom, độ lan nước và tốc độ



Rất chuẩn! Giờ ta sẽ làm 3 loại **item cơ bản nhất** trong game Boom Online:

---

## BƯỚC 10: THÊM 3 LOẠI ITEM (Bom, Nước, Tốc độ)

### Mục tiêu:

- Tạo 3 loại item:
  - **Tăng số lượng bom đặt được** (`"bomb_up"`)
  - **Tăng độ lan nước** (`"range_up"`)
  - **Tăng tốc độ di chuyển** (`"speed_up"`)
- Khi nhân vật chạm vào item → nhận hiệu ứng tương ứng

---

## Các bước cụ thể:

---

### 1. Chuẩn bị ảnh sprite cho 3 item

Giả sử bạn có 1 sprite sheet hoặc 3 hình ảnh:

- ``item_bomb.png`` - tăng số bom
- ``item_range.png`` - tăng độ lan
- ``item_speed.png`` - tăng tốc độ

→ Thêm vào **Atlas** dùng chung (``main.atlas`` hoặc ``items.atlas``) với tên:  
``bomb_up``, ``range_up``, ``speed_up``

---

### 2. Sửa ``item.go`` để hỗ trợ 3 loại:

#### a. Thêm property ``type`` cho item

Trong ``item.script``:

```
go.property("item_type", hash("bomb_up"))
```

#### b. Tùy sprite theo loại item

Trong ``init()``:

```
function init(self)
    sprite.play_flipbook("#sprite", self.item_type)
end
```

---

### 3. Tạo 3 item prototype (prefab):

Bạn có thể tạo 3 file `.go`` riêng biệt hoặc dùng **cùng 1 file** ``item.go``, mỗi factory gán khác nhau:

**Ví dụ: trong** ``main.collection``

- Tạo 3 ``factory``:
  - ``item_bomb_factory``: prototype ``item.go``, set ``item_type = hash("bomb_up")``
  - ``item_range_factory``: prototype ``item.go``, set ``item_type = hash("range_up")``
  - ``item_speed_factory``: prototype ``item.go``, set ``item_type = hash("speed_up")``

(Chỉnh ``item_type`` trong **Properties** của mỗi factory)

---

### 4. Sửa ``bomb.script`` để random item đúng loại:

Thay đoạn rơi item bằng:

```
local rand = math.random()
if rand < 0.33 then
    factory.create("main#item_bomb_factory", p)
elseif rand < 0.66 then
    factory.create("main#item_range_factory", p)
else
    factory.create("main#item_speed_factory", p)
end
```

---

### 5. Nhân vật nhận item đúng hiệu ứng

Trong `player.script` và `bot.script`, thêm:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("item_collected") then
        local item = message.type
        if item == hash("bomb_up") then
            self.max_bombs = (self.max_bombs or 1) + 1
            print("Tăng số bom:", self.max_bombs)
        elseif item == hash("range_up") then
            self.bomb_range = (self.bomb_range or 1) + 1
            print("Tăng độ lan:", self.bomb_range)
        elseif item == hash("speed_up") then
            self.speed = self.speed + 20
            print("Tăng tốc độ:", self.speed)
        end
    end
end
```

📌 Đừng quên khởi tạo `self.max_bombs`, `self.bomb_range` ở `init()` nếu chưa có.

## 6. (Optional) Dùng item trong bomb.script

Nếu bạn muốn **bom nổ lan xa hơn** theo `player.bomb_range`, hãy truyền nó lúc tạo bom.

### ✅ Test Game:

- Phá gạch mềm → rơi 3 loại item ngẫu nhiên
- Nhân vật đi vào → nhận buff
- Có thể print log để thấy hiệu ứng



---

Bạn muốn làm tiếp phần nào:

- 👉 **Hiệu ứng xoay/sáng của item,**
- 👉 **UI hiển thị số bom, tốc độ, nước,**
- 👉 hay **win/lose khi bot hoặc player chết hết?**



ui số bom, nước, tốc độ



Tuyệt vời! Giờ ta sẽ thêm phần **UI hiển thị số bom, độ lan nước và tốc độ** để người chơi biết mình đang mạnh tới đâu rồi.

---

## **BƯỚC 11: THÊM UI CHỈ SỐ (Bom / Nước / Tốc Độ)**

### **Mục tiêu:**

- Hiển thị thông tin người chơi ở góc màn hình:
  - **Số bom đang có**
  - **Độ lan nước**
  - **Tốc độ di chuyển**

---

### **Các bước thực hiện:**

---

## 1. Tạo UI sử dụng `GUI` file

a. Click phải → `New` → `GUI`

Tên: `hud.gui`

(tự động tạo thêm `hud.gui\_script`)

b. Trong `hud.gui`, tạo 3 Text Nodes:

- **Tên node:** `bomb\_text`, `range\_text`, `speed\_text`
- Font: chọn `system\_font.font` hoặc font bạn có
- Vị trí: đặt ở góc trên trái, cách đều nhau

## 2. Hiển thị nội dung ban đầu trong `hud.gui\_script`

```
function init(self)
    gui.set_text(gui.get_node("bomb_text"), "Bom: 1")
    gui.set_text(gui.get_node("range_text"), "Nước: 1")
    gui.set_text(gui.get_node("speed_text"), "Tốc: 60")
end
```

## 3. Gửi thông tin từ `player.script` lên UI

Trong `player.script`, sau khi nhận item:

```
msg.post("main:/hud", "update_ui", {
    bombs = self.max_bombs,
    range = self.bomb_range,
    speed = self.speed
})
```

 Bạn cần set tên game object chứa `hud.gui` là `hud` trong collection.

---

## 4. Nhận và cập nhật UI trong `hud.gui\_script`

Thêm:

```
function on_message(self, message_id, message, sender)
  if message_id == hash("update_ui") then
    gui.set_text(gui.get_node("bomb_text"), "Bom: " .. message.bombs)
    gui.set_text(gui.get_node("range_text"), "Nước: " .. message.range)
    gui.set_text(gui.get_node("speed_text"), "Tốc: " .. message.speed)
  end
end
```

## 5. Thêm `hud.gui` vào scene

- Kéo `hud.gui` vào `main.collection` → tạo `hud.gui` game object
  - Đặt ở layer trên cùng (trên camera/map)
- 

### ✅ Kết quả khi test game:

- Góc trái hiển thị:
  - `Bom: 1`
  - `Nước: 1`
  - `Tốc: 60`

- Khi bạn ăn item → các chỉ số cập nhật **ngay lập tức**

Bạn muốn làm tiếp:

- 👉 **Thanh máu nếu thêm đánh nhau,**
- 👉 **Hiển thị thời gian hoặc số mạng,**
- 👉 Hay làm phần **win/lose UI** khi thắng hoặc thua?



win lose ui đi



Rất hợp lý! Giờ ta sẽ làm **UI Win/Lose** – hiện ra khi người chơi thắng hoặc thua.

## 🏁 BƯỚC 12: THÊM UI WIN / LOSE

### 🎯 Mục tiêu:

- Khi **người chơi hoặc bot chết hết**, hiện màn hình:
  - "YOU WIN" hoặc "YOU LOSE"
- Có thể bấm **phím R để chơi lại**

### 🔧 Các bước thực hiện:

## 1. Cập nhật `hud.gui` để hiển thị trạng thái

Trong `hud.gui`:

- Thêm 1 **Text Node**:
  - Tên: `result\_text`
  - Vị trí: giữa màn hình
  - Text: `"YOU WIN"` hoặc `"YOU LOSE"`
  - Ẩn ban đầu bằng:

```
gui.set_enabled(gui.get_node("result_text"), false)
```

## 2. Cập nhật `hud.gui\_script` để hiển thị kết quả

Thêm hàm:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("update_ui") then
        -- như cũ
        gui.set_text(gui.get_node("bomb_text"), "Bom: " .. message.bombs)
        gui.set_text(gui.get_node("range_text"), "Nước: " .. message.range)
        gui.set_text(gui.get_node("speed_text"), "Tốc: " .. message.speed)

    elseif message_id == hash("game_result") then
        gui.set_text(gui.get_node("result_text"), message.result)
        gui.set_enabled(gui.get_node("result_text"), true)
    end
end
```

### 3. Gửi thông báo thắng/thua

**Trong `player.script`, khi nhân vật chết (ví dụ sau khi bị nước đánh):**

```
msg.post("main:/hud", "game_result", { result = "YOU LOSE" })
```

**Trong `bot.script`, khi bot chết:**

Giả sử bạn chỉ có 1 bot, khi nó chết:

```
msg.post("main:/hud", "game_result", { result = "YOU WIN" })
```

“ Nếu có nhiều bot, bạn cần đếm số bot còn sống.”

### 4. Cho phép chơi lại bằng phím R

Trong `hud.gui\_script` thêm:

```
function on_input(self, action_id, action)
    if gui.is_enabled(gui.get_node("result_text")) and action_id ==
hash("restart") and action.released then
        msg.post("@system:", "restart")
    end
end
```

**Trong `game.input\_binding`:**

- Thêm 1 action:
  - Name: `restart`
  - Input: `keyboard R`

## ✅ Test kết quả:

- Khi bot chết → "YOU WIN" hiện ra
  - Khi người chơi chết → "YOU LOSE"
  - Bấm **R** để chơi lại từ đầu
- 

## 🧠 Có thể nâng cấp sau:

- **Nút Restart trên màn hình** thay vì phím R
  - Hiệu ứng fade in/out
  - Đếm số mạng còn lại, thời gian sống sót
- 

Bạn muốn tiếp theo là gì?

- 👉 Nhiều mạng chơi?
- 👉 Thêm hiệu ứng bom nổ đẹp hơn?
- 👉 Hay tạo **menu chính (Main Menu)**?



menu chính



Tuyệt, giờ ta sẽ làm **Menu Chính** – nơi người chơi bắt đầu game, có nút “Play” và có thể thêm tùy chọn khác như “Quit”, “Instructions”, v.v.

---



## BƯỚC 13: TẠO MENU CHÍNH (MAIN MENU)

---



### Mục tiêu:

- Khi mở game → hiện menu với nút “Play”
  - Bấm “Play” → chuyển sang chơi game chính
  - Tách **2 collection**: ``main_menu`` và ``game``
- 



### Các bước thực hiện:

---

#### 1. Tạo ``main_menu.collection``

- Tạo collection mới → tên: ``main_menu.collection``
  - Kéo vào ``game.project`` phần bootstrap → dùng ``main_menu.collection`` làm collection đầu tiên
- 

#### 2. Tạo GUI menu: ``menu.gui``

##### Gồm:

- 1 **Text Node**: "Boom Online"
- 1 **Box Node** làm nút “Play”
  - Thêm text bên trên: "PLAY"





- Đặt ở giữa màn hình
- Trong ``menu.gui_script``:

```
function init(self)
    self.play_btn = gui.get_node("play_button")
end

function on_input(self, action_id, action)
    if action.released and gui.pick_node(self.play_btn, action.x, action.y)
    then
        -- Chuyển sang game
        msg.post("main:/menu", "start_game")
    end
end
```

### 3. Tạo ``menu.script`` để load game

Tạo 1 game object tên ``menu`` trong ``main_menu.collection``, gắn ``menu.script``.

Trong ``menu.script``:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("start_game") then
        -- Load game collection
        collectionproxy.load("#game_proxy")
    end
end

function on_message(self, message_id, message, sender)
    if message_id == hash("proxy_loaded") then
        collectionproxy.enable("#game_proxy")
    end
end
```

---

## 4. Thêm Collection Proxy để load `game.collection`

- Trong `main\_menu.collection`, game object `menu`:
    - Add `Collection Proxy`
    - Collection: chọn `game.collection`
    - ID: `game\_proxy`
    - Tắt `Auto Load`
- 

### ✅ Test kết quả:

- Mở game → hiện menu
  - Click "PLAY" → vào game chính và chơi như bình thường
- 

### 🧠 Mở rộng sau:

- Nút "Instructions", "Quit"
  - Nhạc nền menu riêng
  - Giao diện chọn map hoặc chọn nhân vật
- 

Bạn muốn làm tiếp gì sau menu?

👉 **Chọn nhân vật,**

- 👉 Thêm nhiều màn chơi,
- 👉 Thêm nhạc & âm thanh?



chọn nhân vật



Tuyệt! Giờ ta sẽ thêm **chức năng chọn nhân vật (character select)** trước khi vào chơi – giống như Boom Online, người chơi được chọn giữa các nhân vật khác nhau.

## BƯỚC 14: THÊM CHỌN NHÂN VẬT

### Mục tiêu:

- Trên **menu chính**, thêm màn hình chọn nhân vật
- Người chơi chọn 1 nhân vật → lưu lại
- Khi vào game, nhân vật được tạo là đúng loại đã chọn

### Các bước thực hiện:

#### 1. Chuẩn bị sprite cho nhiều nhân vật

Ví dụ có 2 nhân vật:

- ``blue.png`` → nhân vật xanh
- ``red.png`` → nhân vật đỏ

→ Thêm cả 2 vào cùng 1 **Atlas** (ví dụ: ``characters.atlas``)

Đặt tên: ``blue``, ``red``

---

## 2. Cập nhật GUI chọn nhân vật (``menu.gui``)

Trong ``menu.gui``, thêm:

- 2 **Box Node** (hoặc Sprite Node): hiển thị ``blue``, ``red``
  - 1 Text Node: “CHỌN NHÂN VẬT”
  - Người dùng **click vào một nhân vật** → lưu lại nhân vật được chọn
- 

## 3. Lưu nhân vật đã chọn

Trong ``menu.gui_script``:

```
function init(self)
    self.selected = "blue"  -- mặc định
    self.blue_node = gui.get_node("blue")
    self.red_node = gui.get_node("red")
end

function on_input(self, action_id, action)
    if action.released then
        if gui.pick_node(self.blue_node, action.x, action.y) then
            self.selected = "blue"
        elseif gui.pick_node(self.red_node, action.x, action.y) then
            self.selected = "red"
        end
    end
end
```

```

elseif gui.pick_node(gui.get_node("play_button"), action.x, action.y)
then
    -- Gửi nhân vật đã chọn sang menu.script
    msg.post("main:/menu", "start_game", { character = self.selected })
end
end
end

```

## 4. Chuyển nhân vật vào `menu.script` và sang game

Trong `menu.script`, lưu lại tên nhân vật và truyền sang game:

```

local selected_character = "blue"

function on_message(self, message_id, message, sender)
    if message_id == hash("start_game") then
        selected_character = message.character
        collectionproxy.load("#game_proxy")
    elseif message_id == hash("proxy_loaded") then
        -- Gửi character vào game sau khi load xong
        msg.post("#game_proxy", "set_character", { character =
selected_character })
        collectionproxy.enable("#game_proxy")
    end
end
end

```

## 5. Nhận trong `game.collection` và spawn đúng nhân vật

Trong `player.script`:

- Thêm `go.property("character", hash("blue"))`

- Trong `init()`, chọn đúng sprite:

```
function init(self)
    sprite.play_flipbook("#sprite", self.character)
end
```

### Trong `main.script` của game:

- Nhận thông điệp:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("set_character") then
        -- Gửi đến player để set sprite
        msg.post("player", "set_character", { character = message.character })
    end
end
```

### Trong `player.script`, thêm:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("set_character") then
        self.character = hash(message.character)
        sprite.play_flipbook("#sprite", self.character)
    end
end
```

---

### ✅ Kết quả khi test:

- Ở menu chính, chọn nhân vật đỏ hoặc xanh
- Nhấn Play → vào game và điều khiển đúng nhân vật

🧠 Bạn có thể nâng cấp:

- Thêm nhiều nhân vật hơn
- Cho hiển thị chỉ số từng nhân vật (tốc độ, sức mạnh...)
- Hiệu ứng chọn (highlight viền khi chọn)

---

Bạn muốn làm tiếp phần nào?

👉 **Bot cũng chọn nhân vật?**

👉 **Tạo nhiều màn chơi?**

👉 **Lưu điểm cao (high score)?**



tạo nhiều màn chơi



Tuyệt! Làm **nhiều màn chơi (multi-level)** giúp game phong phú hơn. Ta sẽ cho người chơi **chọn map** ở menu, rồi khi vào game sẽ **tải đúng map**, ví dụ: Map 1, Map 2, Map 3,...



## **BƯỚC 15: THÊM NHIỀU MÀN CHƠI (MAP/LEVEL)**



### **Mục tiêu:**

- Cho người chơi chọn 1 trong nhiều bản đồ



- Mỗi map là 1 **collection** riêng biệt (hoặc 1 file tilemap)
- Khi vào game → load đúng map được chọn

---

## Cách thực hiện:

---

### ◆ Cách 1: Mỗi map là 1 collection riêng

(đơn giản, hiệu quả với số lượng map nhỏ: 3-5)

---

#### 1. Tạo các bản đồ riêng

- ``map1.collection``
- ``map2.collection``
- ``map3.collection``

Mỗi collection gồm: tilemap, gạch nền, enemy,...

---

#### 2. Tạo GUI chọn màn (``menu.gui``)

Thêm 3 nút “Map 1”, “Map 2”, “Map 3” → người chơi click chọn

Trong ``menu.gui_script``:

```
function init(self)
    self.selected_map = "map1" -- mặc định
end
```



```

function on_input(self, action_id, action)
    if action.released then
        if gui.pick_node(gui.get_node("map1_btn"), action.x, action.y) then
            self.selected_map = "map1"
        elseif gui.pick_node(gui.get_node("map2_btn"), action.x, action.y) then
            self.selected_map = "map2"
        elseif gui.pick_node(gui.get_node("play_button"), action.x, action.y)
    then
        msg.post("main:/menu", "start_game", {
            character = self.selected_character,
            map = self.selected_map
        })
    end
end
end
end

```

### 3. Trong `menu.script`, load đúng map proxy

#### a. Gắn sẵn 3 `collectionproxy` vào game object `menu`:

- `map1\_proxy` → `map1.collection`
- `map2\_proxy` → `map2.collection`
- `map3\_proxy` → `map3.collection`

Tắt **auto-load**

#### b. Trong `menu.script`:

```

local selected_character = "blue"
local selected_map = "map1"

function on_message(self, message_id, message, sender)
    if message_id == hash("start_game") then
        selected_character = message.character
    end
end

```

```

selected_map = message.map

-- Load collection game chính
collectionproxy.load("#game_proxy")
elseif message_id == hash("proxy_loaded") then
    msg.post("#game_proxy", "set_character", { character =
selected_character })

-- Enable map đúng
if selected_map == "map1" then
    collectionproxy.enable("#map1_proxy")
elseif selected_map == "map2" then
    collectionproxy.enable("#map2_proxy")
elseif selected_map == "map3" then
    collectionproxy.enable("#map3_proxy")
end

collectionproxy.enable("#game_proxy")
end
end

```

## ✅ Kết quả:

- Ở menu, bạn chọn Map 1, 2, hoặc 3
- Vào game → đúng map đã chọn được load lên

## 🧠 Nâng cao:

- Dùng **1 tilemap + dữ liệu layout** để tạo map động (Cách 2 – phức tạp hơn)
- Tạo **màn chơi tiếp theo** sau khi thắng (auto load map2 → map3)

- Hiện tên map trong UI (góc trên)

Bạn muốn tiếp theo:

- 👉 Làm **chuyển màn tự động khi thắng?**
- 👉 Hay làm **cơ chế lưu tiến độ qua nhiều màn (save game)?**



chuyển màn tự động



Tuyệt! Bây giờ ta sẽ làm **chuyển màn tự động khi người chơi thắng** — ví dụ: sau khi hạ xong bot ở Map 1 → tự động chuyển qua Map 2.

## BƯỚC 16: CHUYỂN MÀN TỰ ĐỘNG KHI THẮNG

### Mục tiêu:

- Mỗi map là 1 collection riêng biệt: ``map1``, ``map2``, ``map3``
- Khi thắng (ví dụ: bot chết hết) → **tự động load map tiếp theo**
- Nếu hết map → hiện “Bạn đã thắng toàn bộ!”

### Các bước thực hiện:

---

## 1. Theo dõi map hiện tại

Trong `game.collection`, tạo 1 `main.script` để:

- Biết map hiện tại
- Chuyển sang map tiếp theo khi nhận `"next_level"`

---

## 2. Cấu hình các map

Trong `main.script`, tạo danh sách map:

```
local maps = { "map1", "map2", "map3" }  
local current_map_index = 1
```

Giả sử bạn có các proxy:

- `map1_proxy`, `map2_proxy`, `map3_proxy`  
(đặt sẵn trong collection và tắt auto-load)

---

## 3. Xử lý chuyển map

Trong `main.script`:

```
function on_message(self, message_id, message, sender)  
    if message_id == hash("next_level") then  
        current_map_index = current_map_index + 1  
  
        if current_map_index > #maps then  
            msg.post("main:/hud", "game_result", { result = "🏆 Đã thắng toàn" })
```

```

bộ!" })

        return
    end

    -- Unload map cũ
    if current_map_index > 1 then
        collectionproxy.unload("#" .. maps[current_map_index - 1] ..
"_proxy")
    end

    -- Load map mới
    collectionproxy.load("#" .. maps[current_map_index] .. "_proxy")
    elseif message_id == hash("proxy_loaded") then
        collectionproxy.enable(sender)
    end
end
end

```

## 4. Khi người chơi thắng → gửi `next\_level`

Trong `bot.script` hoặc nơi check số lượng bot:

```

if all_bots_dead then
    msg.post("main:/main", "next_level")
end

```

Bạn có thể thêm hiệu ứng delay, ví dụ:

```

timer.delay(1.0, false, function()
    msg.post("main:/main", "next_level")
end)

```

## 5. Hiển thị tên map mới trong `hud.gui` (tùy chọn)

Gửi message từ `main.script`:

```
msg.post("main:/hud", "update_level", { level = current_map_index })
```

Trong `hud.gui\_script`:

```
if message_id == hash("update_level") then
    gui.set_text(gui.get_node("level_text"), "Map: " .. message.level)
end
```

### ✅ Test kết quả:

- Vào Map 1 → thắng → tự load Map 2
- Thắng Map 2 → sang Map 3
- Thắng hết → hiện “🏆 Đã thắng toàn bộ!”

🧠 Nâng cao:

- Thêm **hiệu ứng chuyển màn fade-in/fade-out**
- Hồi máu, reset chỉ số nhẹ giữa màn
- Cho phép **chơi lại từ màn hiện tại**

Bạn muốn làm tiếp gì?

- 👉 Thêm **hiệu ứng chuyển cảnh**?
- 👉 Reset lại chỉ số (bomb/speed) sau mỗi màn?
- 👉 Lưu màn đang chơi để lần sau chơi tiếp?



reset chỉ số sau mỗi màn



Rất hay! Reset chỉ số (số bom, độ lan nước, tốc độ) sau mỗi màn giúp giữ game cân bằng và tăng độ khó dần. Giờ mình sẽ hướng dẫn cách làm.

---

## BƯỚC 17: RESET CHỈ SỐ SAU MỖI MÀN

---

### Mục tiêu:

- Khi **bắt đầu màn mới**, nhân vật sẽ:
  - Có **số bom mặc định**
  - **Độ lan nước mặc định**
  - **Tốc độ chạy mặc định**
- Reset này không ảnh hưởng khi **chết trong cùng 1 màn**

---

### Các bước thực hiện:

---

## 1. Cập nhật chỉ số mặc định trong `player.script`

Thêm biến gốc (mặc định):

```
local DEFAULT_BOMB = 1
local DEFAULT_RANGE = 1
local DEFAULT_SPEED = 100
```

## 2. Thêm hàm reset lại chỉ số

Trong `player.script`, thêm:

```
function reset_stats(self)
    self.bomb_count = DEFAULT_BOMB
    self.range = DEFAULT_RANGE
    self.move_speed = DEFAULT_SPEED

    -- Cập nhật lại UI
    msg.post("main:/hud", "update_ui", {
        bombs = self.bomb_count,
        range = self.range,
        speed = self.move_speed
    })
end
```

## 3. Gọi reset khi load màn mới

Trong `main.script`, sau khi enable map mới:


```
-- Gửi tin nhắn reset tới player
```



```
msg.post("player", "reset_stats")
```

Trong `player.script`, xử lý message:

```
function on_message(self, message_id, message, sender)
    if message_id == hash("reset_stats") then
        reset_stats(self)
    end
end
```

“ Bạn có thể gọi `reset_stats(self)` cả trong `init()` để đảm bảo trạng thái ban đầu.”

## ✅ Kết quả khi test:

- Khi vào **màn đầu tiên** → chỉ số mặc định
- Nhặt item → chỉ số tăng
- **Thắng màn** → **sang màn mới** → chỉ số reset lại

## 🧠 Nâng cao (tùy chọn):

- Tăng độ khó theo màn (ví dụ: map2 → bot mạnh hơn)
- Cho giữ lại 1 số chỉ số nếu muốn thiết kế game kiểu "phát triển dần"
- Hiện **"Đã reset chỉ số!"** ở góc màn hình vài giây

Bạn muốn làm gì tiếp?

- 👉 Thêm **hiệu ứng chuyển màn (fade)**?
- 👉 Làm **nhiều loại bot** thông minh hơn?
- 👉 Hay bắt đầu làm **chế độ chơi 2 người**?



bot thông minh hơn



Tuyệt! Giờ ta sẽ nâng cấp **AI bot thông minh hơn** – thay vì chỉ đi ngẫu nhiên, bot sẽ biết:

- ✅ **Tránh nước bom**
- ✅ **Tìm đường tới player**
- ✅ **Đặt bom khi đến gần player**

## BƯỚC 18: BOT THÔNG MINH HƠN

### Mục tiêu:

- Nếu phát hiện có **nguy hiểm (nước bom)** → tìm đường **tránh né**
- Nếu gần **người chơi** → tiến đến gần, **đặt bom**
- Di chuyển không quá "điên loạn"

### Giai đoạn 1: Bot tránh nước bom

---

## 1. Xác định vùng nguy hiểm

Trong `bot.script`, mỗi frame kiểm tra nếu ô kế bên có nước bom:

```
function is_danger_near(pos)
  -- Kiểm tra các hướng lân cận
  local directions = {
    vmath.vector3(1, 0, 0),
    vmath.vector3(-1, 0, 0),
    vmath.vector3(0, 1, 0),
    vmath.vector3(0, -1, 0)
  }

  for _, dir in ipairs(directions) do
    local check_pos = pos + dir * TILE_SIZE
    local tile = tilemap.get_tile("#tilemap", "water",
math.floor(check_pos.x / TILE_SIZE), math.floor(check_pos.y / TILE_SIZE))
    if tile ~= 0 then
      return true
    end
  end

  return false
end
```

---

## 2. Nếu có nước bom gần → chạy ngược hướng

Trong update hoặc AI tick:

```
if is_danger_near(self.position) then
  -- Lùi ra xa, đơn giản là đổi hướng ngẫu nhiên không dính nước
  local safe_dirs = get_safe_directions(self.position)
  if #safe_dirs > 0 then
```

```
        self.dir = safe_dirs[math.random(1, #safe_dirs)]
    end
end
```

## Giai đoạn 2: Bot tiến đến player & đặt bom

### 3. Lấy vị trí player

Giả sử trong `bot.script`:

```
local player_pos = go.get_position("player")

-- Nếu ở gần player (cùng ô hoặc cạnh)
if vmath.length(player_pos - self.position) < TILE_SIZE + 1 then
    -- Đặt bom
    if not self.bomb_cooldown then
        msg.post("#bomb_factory", "create", { position = self.position })
        self.bomb_cooldown = true
        timer.delay(2.0, false, function() self.bomb_cooldown = false end)
    end
end
```

## Giai đoạn 3: Hướng thông minh (tìm đường)

Nếu bạn muốn *\*bot tìm đường thật sự (A)\*\**, ta cần thêm:

- **Biểu đồ map** dạng lưới (grid logic)
- Viết hàm **tìm đường ngắn nhất** từ A đến B

- Chuyển hướng bot theo đường đã tìm

“🧱 *Cái này có thể làm sau nếu bạn muốn AI nâng cao thực sự!*”

## ✅ Tóm tắt:

Tính năng	Đã thêm?
Tránh vùng nước bom	✅ Có
Tìm đến người chơi	✅ Đơn giản
Đặt bom khi gần player	✅ Có
Tìm đường thông minh A*	➡️ <small>SOON</small> Có thể thêm sau

Bạn muốn nâng tiếp phần nào?

👉 Làm **bot đặt bom phá gạch?**

👉 Làm **bot có cấp độ khó khác nhau?**

👉 Hay làm luôn *AI pathfinding bằng A?\**



bot đặt bom phá gạch



Tuyệt! Ta sẽ nâng cấp để **bot biết đặt bom phá gạch mềm** — giúp nó tự mở đường, không bị kẹt, và tạo nguy hiểm cho người chơi.



## BƯỚC 19: BOT ĐẶT BOM PHÁ GẠCH MỀM



### Mục tiêu:

- Khi gặp **gạch mềm** chắn đường → bot sẽ **đặt bom để phá**
- Đợi nước bom dứt rồi mới đi tiếp
- Không spam bom liên tục (cooldown)



### Các bước thực hiện:

#### 1. Xác định gạch mềm gần bot

Trong `bot.script``:

```
function is_soft_block_near(pos)
    local dirs = {
        vmath.vector3(1, 0, 0),
        vmath.vector3(-1, 0, 0),
        vmath.vector3(0, 1, 0),
        vmath.vector3(0, -1, 0),
    }

    for _, dir in ipairs(dirs) do
        local check = pos + dir * TILE_SIZE
        local tile_x = math.floor(check.x / TILE_SIZE)
        local tile_y = math.floor(check.y / TILE_SIZE)

        local tile = tilemap.get_tile("#tilemap", "softblock", tile_x, tile_y)
```



```
        if tile ~= 0 then
            return true
        end
    end

    return false
end
```

## 2. Đặt bom nếu có gạch mềm gần và chưa cooldown

Thêm logic vào update hoặc AI tick:

```
if not self.bomb_cooldown and is_soft_block_near(self.position) then
    msg.post("#bomb_factory", "create", { position = self.position })

    self.bomb_cooldown = true
    timer.delay(2.0, false, function()
        self.bomb_cooldown = false
    end)
end
```

## 3. (Tùy chọn) Ưu tiên gạch chắn đường

Bạn có thể dùng raycast tile hoặc line-of-sight đơn giản để xác định:

*“Tôi đang bị gạch mềm chắn đường?” → đặt bom phá”*

 **Kết quả:**

- Bot gặp gạch mềm → đặt bom phá
  - Đợi bom nổ xong → tiếp tục di chuyển
  - Tự mở đường tìm đến player
- 

## **Nâng cao:**

- **Tính toán hướng tẩu thoát** trước khi đặt bom
  - Cho bot chỉ phá gạch nếu **đang bị kẹt**
  - Bot cấp độ cao hơn có thể **“dụ player vào bom”**
- 

Bạn muốn bot khôn hơn nữa?

👉 Làm *\*bot dò đường (A)\**?

👉 Làm **bot biết trốn sau khi đặt bom**?

👉 Hay bắt đầu làm **chế độ chơi 2 người**?



bot biết trốn



Rất tốt! Giờ ta sẽ nâng cấp bot để **biết trốn sau khi đặt bom**, tránh bị tự nổ — đây là bước quan trọng để bot thông minh và giống người chơi thật.

---





## BƯỚC 20: BOT TRỐN SAU KHI ĐẶT BOM



### Mục tiêu:

- Sau khi bot **đặt bom**, nó **chạy ra khỏi vùng nổ**
- Không quay lại vùng nước bom cho đến khi an toàn
- Không bị kẹt lại tại vị trí đặt bom



### Các bước thực hiện:

#### 1. Đặt biến trạng thái “đang trốn”

Trong `bot.script`:

```
self.escaping = false
self.escape_target = nil
```

#### 2. Sau khi đặt bom → tìm ô an toàn để trốn

```
function find_safe_tile(pos)
    local dirs = {
        vmath.vector3(1, 0, 0),
        vmath.vector3(-1, 0, 0),
        vmath.vector3(0, 1, 0),
        vmath.vector3(0, -1, 0)
    }
```



```

}

for _, dir in ipairs(dirs) do
    local check = pos + dir * TILE_SIZE
    local tile_x = math.floor(check.x / TILE_SIZE)
    local tile_y = math.floor(check.y / TILE_SIZE)

    -- không có nước, không có gạch
    local water = tilemap.get_tile("#tilemap", "water", tile_x, tile_y)
    local block = tilemap.get_tile("#tilemap", "collision", tile_x, tile_y)

    if water == 0 and block == 0 then
        return check
    end
end

return nil
end

```

### 3. Sau khi đặt bom → bật chế độ trốn

Trong đoạn đặt bom:

```

if not self.bomb_cooldown and is_soft_block_near(self.position) then
    msg.post("#bomb_factory", "create", { position = self.position })

    self.bomb_cooldown = true
    timer.delay(2.0, false, function()
        self.bomb_cooldown = false
    end)

    -- Kích hoạt escape mode
    self.escaping = true
    self.escape_target = find_safe_tile(self.position)
end

```

---

## 4. Di chuyển tới vùng an toàn

Trong `update()` hoặc AI tick:

```
if self.escaping and self.escape_target then
    local dir = vmath.normalize(self.escape_target - self.position)
    self.velocity = dir * self.move_speed

    -- Khi đã đến nơi trốn
    if vmath.length(self.escape_target - self.position) < 1 then
        self.velocity = vmath.vector3()
        self.escaping = false
        self.escape_target = nil
    end
end
```

Đừng quên cập nhật `go.set_position(...)` hoặc dùng `go.animate()` để di chuyển bot.

---

## ✅ Kết quả:

- Bot đặt bom phá gạch hoặc tấn công → **tự động tìm đường trốn**
  - Không bị nổ chính bom của nó
  - Đợi vùng nổ hết rồi mới quay lại
- 

## 🧠 Nâng cao:

- Dùng thuật toán A\* để tìm đường trốn xa hơn nếu gần hết chỗ

- Tránh bom của chính **người chơi**
- Phân loại bot thông minh/trung bình/ngốc 😏

Bạn muốn nâng cấp tiếp?

👉 Làm **AI tìm đường A\***?

👉 Cho bot **sử dụng item**?

👉 Hay làm **chế độ 2 người chơi (VS)**?



--[



Có vẻ bạn vừa gõ `--[` — đây thường là bắt đầu một đoạn comment dài trong Lua.

Ví dụ đúng của comment block trong Lua là:

```
-- Đây là comment 1 dòng

--[[
    Đây là comment nhiều dòng
    Dùng để giải thích đoạn code dài
]]
```

Bạn muốn viết comment block, hay bạn định gõ gì khác?



làm sao để tôi tải xuống toàn bộ cuộc trò chuyện này



Hiện tại, **ChatGPT chưa có nút chính thức để tải toàn bộ cuộc trò chuyện dưới dạng file (PDF, TXT...)** trực tiếp, nhưng bạn có thể dùng một trong các cách sau để lưu lại:

---

### ✅ **Cách 1: Dùng tính năng "Print to PDF" trên trình duyệt**

1. **Nhấn Ctrl + P** (Windows) hoặc **Cmd + P** (Mac) trên trang này.
2. Trong cửa sổ in, chọn **"Save as PDF"** hoặc **"In ra tệp PDF"**.
3. Nhấn **Save / Lưu** để tải về.

💡 *Tip: Bạn có thể cuộn lên đầu cuộc trò chuyện để in toàn bộ.*

---

### ✅ **Cách 2: Thủ công copy - paste vào Word hoặc Notepad**

1. Cuộn lên đầu đoạn chat.
  2. Chọn toàn bộ bằng chuột, **Ctrl + C** để sao chép.
  3. Mở ứng dụng Word, Notepad, hoặc Google Docs.
  4. **Dán (Ctrl + V)** rồi **lưu lại**.
- 

### ✅ **Cách 3: Dùng tiện ích trình duyệt (nâng cao)**

Nếu bạn thường xuyên cần lưu hội thoại:

- Dùng extension như **ChatGPT Exporter**, **ChatGPT Saver** (có trên Chrome Web Store).
- Các tiện ích này cho phép lưu dạng **Markdown, PDF, JSON...**

Bạn muốn mình **tạo bản tổng hợp nội dung toàn bộ hướng dẫn làm game Boom Online này** dưới dạng file để bạn tải luôn không?



có



Đã phân tích ^

```
from datetime import datetime
from pathlib import Path

# Tổng hợp nội dung dạng text
conversation_summary = """
🔥 HƯỚNG DẪN LÀM GAME PHONG CÁCH BOOM ONLINE (CRAZY ARCADE) BẰNG DEFOLD

✅ Bản đồ`
- Tạo map bằng tilemap: tường cứng, gạch mềm, nền.
- Sử dụng các tile layer để xử lý va chạm và hiển thị.

✅ Nhân vật di chuyển
- Điều khiển bằng phím (WASD hoặc mũi tên).
- Xử lý va chạm với gạch, tường.

✅ Đặt bom bằng SPACE
- Bom nổ sau 2s.
- Nước lan 4 hướng, phá gạch mềm, hạ nhân vật.

✅ AI bot cơ bản
```

- Di chuyển ngẫu nhiên, tránh tường.

#### ✅ AI tránh bom

- Bot phát hiện vùng nước bom và tự động tránh xa.

#### ✅ Item

- 3 loại: tăng số bom, tăng độ lan nước, tăng tốc độ.
- Khi nhân vật chạm item → cập nhật chỉ số & gửi tin nhắn update UI.

#### ✅ UI

- Góc trên hiện số bom, nước, tốc độ.
- Màn hình Win / Lose khi thắng hoặc thua.

#### ✅ Menu chính

- Nút bắt đầu game, chọn nhân vật.

#### ✅ Chọn nhân vật

- Mỗi nhân vật có hình ảnh riêng, có thể gán ID nếu cần logic riêng.

#### ✅ Nhiều màn chơi

- Map được load động từ nhiều file .tilemap khác nhau.
- Khi Win → tự động chuyển sang map kế tiếp.

#### ✅ Reset chỉ số mỗi màn

- Khi vào màn mới → chỉ số bomb/range/speed trở về mặc định.

#### ✅ AI thông minh hơn

- Tránh nước bom.
- Tìm đường đến player.
- Đặt bom khi gần player.

#### ✅ Bot đặt bom phá gạch

- Khi gặp gạch mềm chắn đường → đặt bom để phá.

#### ✅ Bot biết trốn sau khi đặt bom

- Sau khi đặt bom → tìm ô an toàn và chạy đến đó.
- Không quay lại vùng nguy hiểm cho đến khi an toàn.

#### 📝 Tổng kết:

- Game Boom Online cơ bản đã hoạt động đầy đủ logic gameplay.

- Có thể mở rộng thêm chế độ 2 người, nhiều bot, hoặc PvP online.

"""

*# Tạo file TXT để tải*

```
file_path = Path("/mnt/data/huong_dan_game_boom_online_defold.txt")
```

```
file_path.write_text(conversation_summary, encoding="utf-8")
```


```
file_path.name
```

Kết quả

```
'huong_dan_game_boom_online_defold.txt'
```

✅ Mình đã tạo file tổng hợp toàn bộ hướng dẫn làm game Boom Online bằng Defold.

Bạn có thể tải tại đây:

 [huong\\_dan\\_game\\_boom\\_online\\_defold.txt](#)

Bạn có muốn mình tạo thêm bản **PDF** hoặc **Markdown** không? [↗]