

第4章

Gitでバージョン管理

～ブランチ編～

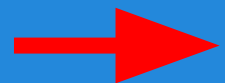
私「新機能～実装♪…まだ適用しないんだった。」
Git「余裕の並行作業、望む時に適用するがよい。」

今回の目的

- ブランチの作成 ([branch](#))
作業用のブランチを作成して、このブランチに対してコミットします。他のブランチには一切影響を与えないので、自由にコミットができます。
- マージの適用 ([merge](#))
現在のブランチに、別ブランチの変更を適用します。差分だけが適用された形でコミットされます。

グラフ	説明
	master [add]画面外に出たら加速度をゼロに設定
	[add]ジャンプ力の範囲を3から10までに制限を追加
	[modify]右に向かってジャンプするように修正
	[update]Unity 5.4.1f1
	[add]Playerスクリプトにジャンプを実装。左クリックと右クリックに対応。
	[add]Playerスクリプトを作成して、Cubeにアタッチ
	[add]CubeにRigidbodyをアタッチ
	[add]Cubeをシーンに作成
	init

別ブランチ



マージ



グラフ	説明
	master Merge branch 'stage1'
	stage1 [add] ランダムな位置に敵を配置
	[add] 追従カメラを実装
	[add]床と天井を作成
	[add]画面外に出たら加速度をゼロに設定
	[add]ジャンプ力の範囲を3から10までに制限を追加
	[modify]右に向かってジャンプするように修正
	[update]Unity 5.4.1f1

パス名順	
+	Assets/AutoCamera.cs
+	Assets/AutoCamera.cs.meta
+	Assets/EnemySpawner.cs
+	Assets/EnemySpawner.cs.meta
...	Assets/Main.unity

Merge branch 'stage1'	
* stage1:	[add] ランダムな位置に敵を配置
	[add] 追従カメラを実装
	[add]床と天井を作成

実施内容

- 別ブランチ(stage1)を作成する！
- 別ブランチで実装とコミットを3回する！
- メインブランチ(master)に3回分の変更をマージで適用する！

ブランチの作成 (*branch*)

1. masterブランチを選択して、ブランチボタンをクリックします。



ブランチの作成 (*branch*)

2. 新規ブランチに“stage1”と入力して、ブランチを作成します。

新規ブランチ ブランチを削除

現在のブランチ: master

新規ブランチ: stage1

コミット: 作業コピーの親
 指定のコミット: 選択...

新規ブランチをチェックアウト

キャンセル ブランチを作成

ブランチの作成 (*branch*)

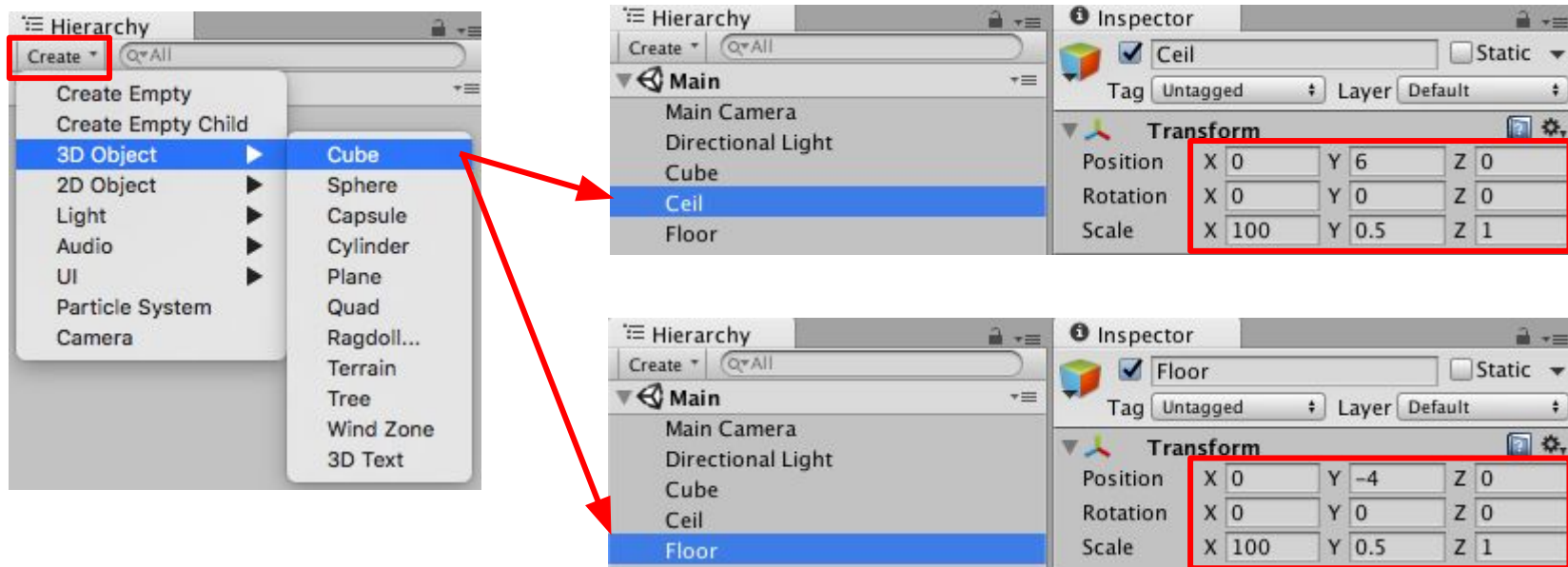
3. stage1ブランチがmasterブランチと同じ位置に作成されたことを確認します。



4. stage1ブランチが現在のブランチになっていることを確認します。

コミット1 (床と天井の作成)

1. Cubeを2つ作成して、Floor(床)とCeil(天井)のTransformを設定します。



コミット1 (床と天井の作成)

2. Unityのシーンをセーブして、stage1にコミットします。



コミット1 (床と天井の作成)

3. stage1にコミットされたことを確認します。
(masterより1コミット先行しています。)

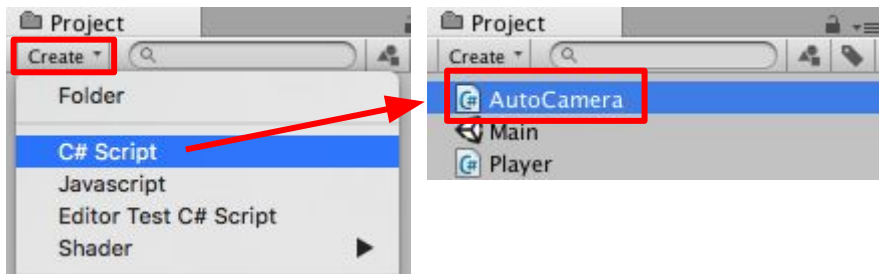


	グラフ	説明
ファイルステータス		
履歴		
検索		
ブランチ		
master		
stage1		

stage1 [add]床と天井を作成
master [add]画面外に出たら加速度をゼロに設定
[add]ジャンプ力の範囲を3から10までに制限を追加
[modify]右に向かってジャンプするように修正
[update]Unity 5.4.1f1
[add]Playerスクリプトにジャンプを実装。左クリックと右クリックに...
[add]Playerスクリプトを作成して、Cubeにアタッチ
[add]CubeにRigidbodyをアタッチ

コミット2 (追従カメラの実装)

1. AutoCamera.csを作成します。



2. AutoCameraを実装します。
(カメラがTarget追従する機能)

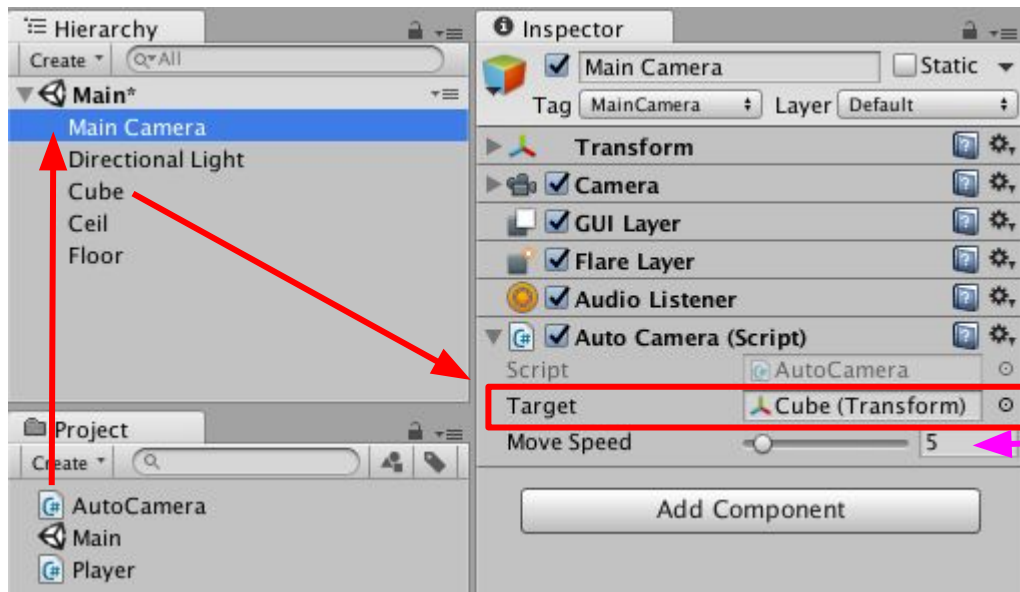
カメラ位置からTarget位置に向けて
x座標のみをスムーズに追従させる

```
1 + using UnityEngine;
2 + using System.Collections;
3 +
4 + public class AutoCamera : MonoBehaviour {
5 +
6 +     public Transform Target;
7 +
8 +     [Range(0f, 60f)]
9 +     public float MoveSpeed = 5f;
10 +
11 +     void Update () {
12 +         var pos = transform.position;
13 +
14 +         pos.x = Mathf.Lerp(
15 +             transform.position.x,
16 +             Target.position.x,
17 +             Time.smoothDeltaTime * MoveSpeed);
18 +
19 +         transform.position = pos;
20 +     }
21 + }
```

カメラの
追従速度

コミット2 (追従カメラの実装)

3. AutoCameraをカメラにアタッチします。
4. アタッチしたAutoCameraのTargetにCubeを設定します。



MoveSpeedが
0だと追従しない、
60だと瞬間移動で
追従する

コミット2 (追従カメラの実装)

5. Unityのシーンをセーブして、stage1にコミットします。

The screenshot shows a Git GUI interface with the following elements:

- File Staging Area:** A list of files to be committed, including `Assets/AutoCamera.cs`, `Assets/Main.unity`, and their meta files. A red box highlights this area.
- Code Editor:** Displays the C# code for `AutoCamera.cs`, showing a `Move` method that moves a camera towards a target.
- Commit Message:** The message `[add] 追従カメラを実装` is entered in the commit message field, highlighted with a red box.
- Commit Action:** The `コミット` button is highlighted with a red box.

Red arrows indicate the flow from the staged files to the commit message and then to the commit button.

コミット2 (追従カメラの実装)

6. stage1にコミットされたことを確認します。
(masterより2コミット先行しています。)

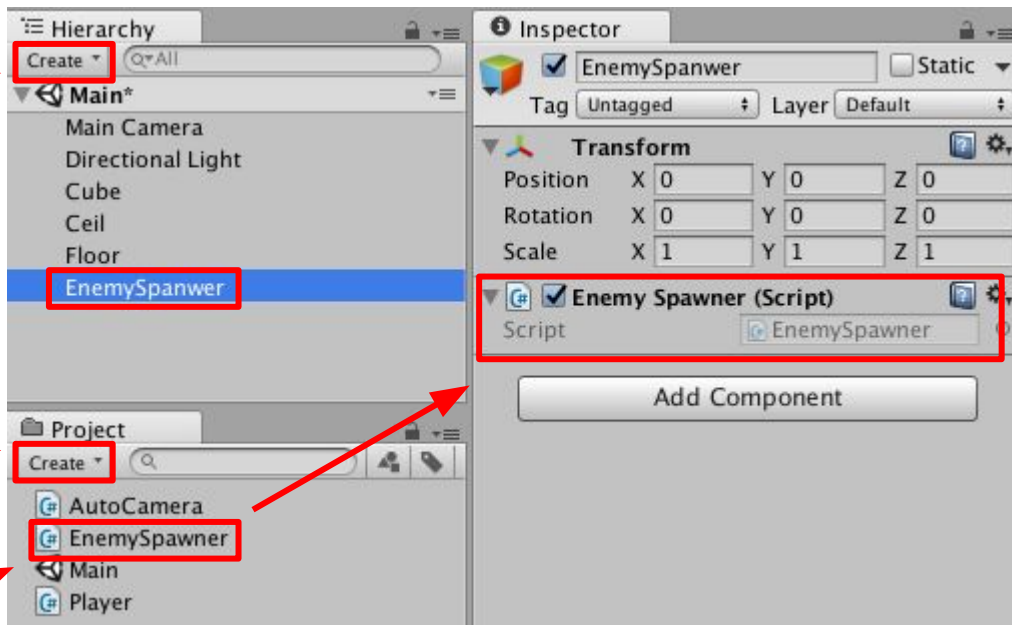
ファイルステータス	グラフ	説明
履歴	○	stage1 [add] 追従カメラを実装
検索	●	[add]床と天井を作成
ブランチ	●	master [add]画面外に出たら加速度をゼロに設定
master	●	[add]ジャンプ力の範囲を3から10までに制限を追加
stage1	●	[modify]右に向かってジャンプするように修正
	●	[update]Unity 5.4.1f1
	●	[add]Playerスクリプトにジャンプを実装。左クリックと右クリックに...
	●	[add]IPlayerスクリプトを作成して、Cubeにアタッチ

コミット3 (敵配置の実装)

1. Create Emptyを選択して
EnemySpawner
ゲームオブジェクトを
作成します。

2. C# Scriptを選択して、
EnemySpawner.csを
作成します。

3. EnemySpawnerを
アタッチします。



コミット3 (敵配置の実装)

4. EnemySpawnerを実装します。
(初期化時に敵(シリンダー形状)をランダムで配置します。)

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class EnemySpawner : MonoBehaviour {
5
6     void Start () {
7         var x = 2f;
8         while (x < 50)
9         {
10             var enemy = GameObject.CreatePrimitive(PrimitiveType.Cylinder);
11             enemy.transform.position = new Vector3(x, Random.Range(-3f, 5f));
12             x += 4f;
13         }
14     }
15 }
```

コミット3 (敵配置の実装)

5. Unityのシーンをセーブして、stage1にコミットします。

The screenshot shows a Git GUI interface. On the left, the 'ワークスペース' (Workspace) sidebar is visible, with 'stage1' selected. The 'Index にステージしたファイル' (Staged files) list is highlighted with a red box and contains the following items:

- Index にステージしたファイル
- Assets/EnemySpawner.cs
- ../EnemySpawner.cs.meta
- Assets/Main.unity

The main editor area shows the content of 'Assets/EnemySpawner.cs', which is a C# script for an enemy spawner. The code is as follows:

```
1 + using UnityEngine;
2 + using System.Collections;
3 +
4 + public class EnemySpawner : MonoBehaviour {
5 +
6 +     void Start () {
7 +         var x = 2f;
8 +         while (x < 50)
9 +             {
10 +                 var enemy = GameObject.CreatePrimitive(PrimitiveType.
11 +                 enemy.transform.position = new Vector3(x, Random.Rang
12 +                 x += 4f;
13 +             }
14 +         }
15 + }
```

At the bottom, the commit message field contains '[add] ランダムな位置に敵を配置', which is also highlighted with a red box. The 'コミット' (Commit) button is highlighted with a red box. A red arrow points from the 'Index にステージしたファイル' list to the commit message field, and another red arrow points from the commit message field to the 'コミット' button.

コミット3 (敵配置の実装)

6. stage1にコミットされたことを確認します。
(masterより3コミット先行しています。)

The screenshot shows a Git commit history interface. On the left, there is a sidebar with navigation options: 'ファイルステータス', '履歴', '検索', 'ブランチ', 'master', and 'stage1'. The 'stage1' branch is selected and highlighted with a red box. In the center, a vertical timeline labeled 'グラフ' shows a series of commit nodes, with the top node highlighted in blue. On the right, the '説明' (Description) column lists commit messages. The top commit, on the 'stage1' branch, is highlighted with a red box and reads: 'stage1 [add] ランダムな位置に敵を配置'. Below it are other commits on the 'master' branch, including '追従カメラを実装', '床と天井を作成', '画面外に出たら加速度をゼロに設定', 'ジャンプ力の範囲を3から10までに制限を追加', '右に向かってジャンプするように修正', 'Unity 5.4.1f1 update', and a commit about implementing jump in a player script.

ブランチ	説明
stage1	[add] ランダムな位置に敵を配置
master	[add] 追従カメラを実装
master	[add] 床と天井を作成
master	[add] 画面外に出たら加速度をゼロに設定
master	[add] ジャンプ力の範囲を3から10までに制限を追加
master	[modify] 右に向かってジャンプするように修正
master	[update] Unity 5.4.1f1
master	[add] 1Player スクリプトにジャンプを実装。左クリックと右クリックに...

マージの適用 (*merge*)

1. masterをダブルクリックして、現在のブランチをmasterにします。
2. マージをクリックします。



マージの適用 (merge)

- stage1(最新のコミット)を選択します。
- オプションをチェックします。
(今回は結果が分かりやすいように設定しています。)
- OKをクリックします。

現在のツリーにマージするコミットを選択して下さい:

すべてのブランチ リモートブランチを表示 親子関係で並べ替え ジャンプ先:

グラフ	説明	コミット	作者	日時
[stage1]	[add] ランダムな位置に敵を配置	fc1cdf6	sab <shizuoka@a...	今日 23:05
	[add] 追加カマフを実装	6481336	sab <shizuoka@a...	今日 22:36
	[add] 床と天井を作成	fc9c8de	sab <shizuoka@a...	今日 22:04
[master]	[add] 画面外に出たら加速度をゼロに設定	e944821	sab <shizuoka@a...	2016/09/15 23:...
	[add] ジャンプ力の範囲が3から10までに制限を追加	e2a6930	sab <shizuoka@a...	2016/09/15 23:27

パス名限

Assets/EnemySpawner.cs

Assets/EnemySpawner.cs.meta

Assets/Main.unity

[add] ランダムな位置に敵を配置

コミット: fc1cdf644a370997fa7f90ec264fad581e586b6e [fc1cdf6]
親: 64813365...

オプション

- マージ結果を直ちにコミット (競合がない場合のみ)
- マージコミットでマージされるコミットからメッセージを読み込む
- fast-forward 可能でも新たなコミットを作成する
- マージではなくリベースする (警告: 変更をプッシュしていないことを確かめて下さい)

キャンセル OK

ブランチのマージ完了

stage1(3回のコミット)の変更が、
masterに反映されました。

masterブランチと分けることで
stage1の実装作業が中途半端でも、
いつでもmasterの状態に戻って
バグ修正することもできます。
複数の別ブランチを作成することも
できます。

The screenshot shows a Git GUI interface. On the left, a sidebar lists 'ワークスペース' (Workspace) with sub-items: 'ファイルステータス' (File status), '履歴' (History), '検索' (Search), 'ブランチ' (Branches), 'タグ' (Tags), 'リモート' (Remote), '一時退避' (Stash), 'サブモジュール' (Submodules), and 'サブツリー' (Subtrees). The 'ブランチ' section is expanded, showing 'master' (selected) and 'stage1'. The main area displays a commit graph with a merge operation from 'stage1' to 'master'. The commit message for the merge is 'Merge branch 'stage1''. Below the graph, a list of files is shown with their status: 'Assets/AutoCamera.cs', 'Assets/AutoCamera.cs.meta', 'Assets/EnemySpawner.cs', 'Assets/EnemySpawner.cs.meta', and 'Assets/Main.unity'. A detailed view of the merge operation is shown at the bottom, including the commit icon and the message 'Merge branch 'stage1'' followed by the commit details for 'stage1': '[add] ランダムな位置に敵を配置', '[add] 追従カメラを実装', and '[add] 床と天井を作成'.

補足

- マージ時の競合 (コンフリクト)
別ブランチ(相手)を現在のブランチ(自分)にマージするとき、互いに同じ箇所を変更していると競合が発生します。相手と自分のどちらの変更を反映するか判断できないためです。
- 並行作業のファイルを分担
複数のブランチで同じファイルを変更しないようにして、競合を防ぎましょう。担当ファイルを分けることが最も安全です。
- 競合が解決しにくいファイル
ソースコードなどのテキストファイルは、競合が発生した行を手動編集して解決できます。シーンファイルや画像ファイルなどは、相手か自分のどちらかで反映する必要があります。(行単位ではなくファイル単位で解決)

まとめ

- ブランチの作成 ([branch](#))
同時にいくつもの並行作業ができます。マージするまでは別ブランチに影響を与えません。
- マージの適用 ([merge](#))
別ブランチ(コミット)の変更を反映できます。
複数のブランチで並行作業したものを、好きなタイミングで必要なものだけ反映することができます。

第4章 クリア

最強無敵のGitでブランチを作成して他に影響を与えずに作業ができました。検討作業や修正作業などブランチを分けて、遠慮なく変更していきましょう。

次回は、ログ・リバート・リセットを使って、履歴の確認や打ち消しを試してみましょう。