

第2章

Gitでバージョン管理

～コミット修正編～

私「コミット(カタカタ…ターン)！あ…ミスってた。」
Git「お主の歴史は自由に直してやるぞよ。」

今回の目的

- ブランチを作成 ([branch](#))
バックアップ目的でブランチを作成します。
初心者におすすめの活用法です！
- コミットを取消 ([reset](#))
一度行ったコミットを取消して、過去に戻ります。
- 直前のコミットを修正 ([amend](#))
メッセージの誤字やコミット漏れを修正します。

すべてのブランチ リモートブランチを表示 日時の順

ファイルステータス: 作業コピー

ブランチ

- master

タグ

リモート

樹形図

| | |
|--------|---------------------------------|
| master | [add]Playerスクリプトにジャンプを実装 |
| | [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 |

「右クリックに対応」を普通にコミットする。
※tmpというブランチも作成

樹形図

| | |
|--------|-----------------------------------|
| master | [modify]Playerスクリプトのジャンプで右クリックに対応 |
| tmp | [add]Playerスクリプトにジャンプを実装 |
| | [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 |

一度行ったコミットを修正したい。
「右クリックのジャンプを実装し忘れた！」

「右クリックに対応」のコミットを取消。

すべてのブランチ リモートブランチを表示 日時の順

樹形図

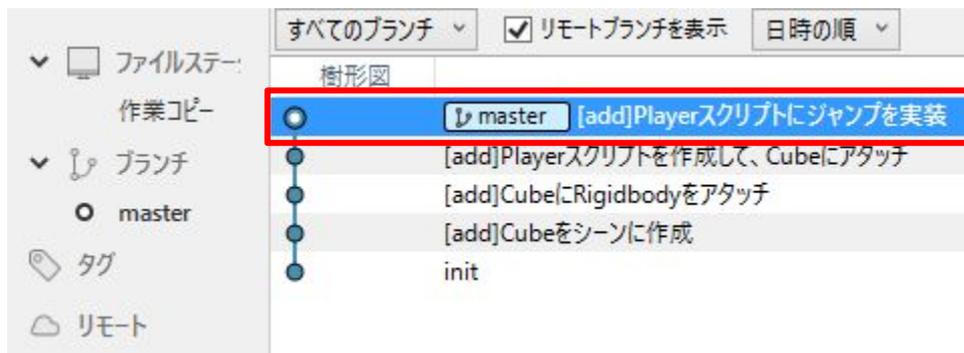
| | |
|--------|--|
| master | [add]Playerスクリプトにジャンプを実装。左クリックと右クリックに対応。 |
| | [add]Playerスクリプトを作成して、Cubeにアタッチ |
| | [add]CubeにRigidbodyをアタッチ |
| | [add]Cubeをシーンに作成 |
| | init |

樹形図

| | |
|--------|---------------------------------|
| | コミットされていない変更があります |
| master | [add]Playerスクリプトにジャンプを実装 |
| tmp | [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 |

ブランチを作成 (*branch*)

- 本来ブランチは、並行作業をするために使用します。
 - 今回は、バックアップ目的にブランチを作成します。
1. masterブランチの最新コミットを選択します。
 2. 右クリックして”ブランチ…”を選択します。(ブランチボタンでも可)



ブランチを作成 (branch)

3. ブランチ名を入力し、チェックアウトをOFFにします。

チェックアウトON: tmpブランチが現在のブランチになります。

チェックアウトOFF: masterブランチのまま変わりません。

tmpはバックアップ目的のため、チェックアウトする必要はありません。

ブランチ

新規ブランチ

現在のブランチ: master

新規ブランチ: tmp

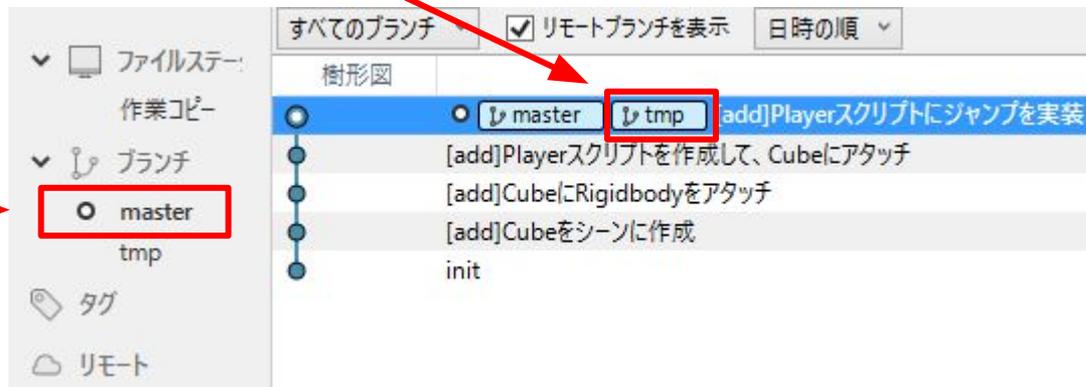
コミット: 作業コピーの親 指定のコミット: 294603ca150dabea91d1d941 ...

新規ブランチを作成してチェックアウト

ブランチを作成 (branch)

4. tmpブランチが作成されたことを確認します。

5. masterブランチが
アクティブで
あることを
確認します。



ブランチを作成 (*branch*)

- UnityのプロジェクトビューからPlayer.csを開きます。
- 以下のコードのように右クリックでジャンプできるように修正します。

```
11 [Tooltip("ジャンプする高さ")]
12 public float Jump = 5f;
13
14 private bool _isJump;
15
16 void Awake()
17 {
18     _transform = GetComponent<Transform>();
19     _rigidbody = GetComponent<Rigidbody>();
20 }
21
22 void Update()
23 {
24     _isJump = _isJump || Input.GetMouseButtonDown(0) || Input.GetMouseButtonDown(1);
25 }
```



- Unityを再生し、左クリックと右クリックでジャンプできることを確認します。

ブランチを作成 (*branch*)

9. 修正したPlayer.csをコミットします。図のようになります。



masterブランチが1つ進み、
tmpブランチが元の位置に残ります。

tmpブランチがある限り、
いつでも元の状態に戻れます。
つまり、**バックアップの効果**があります。

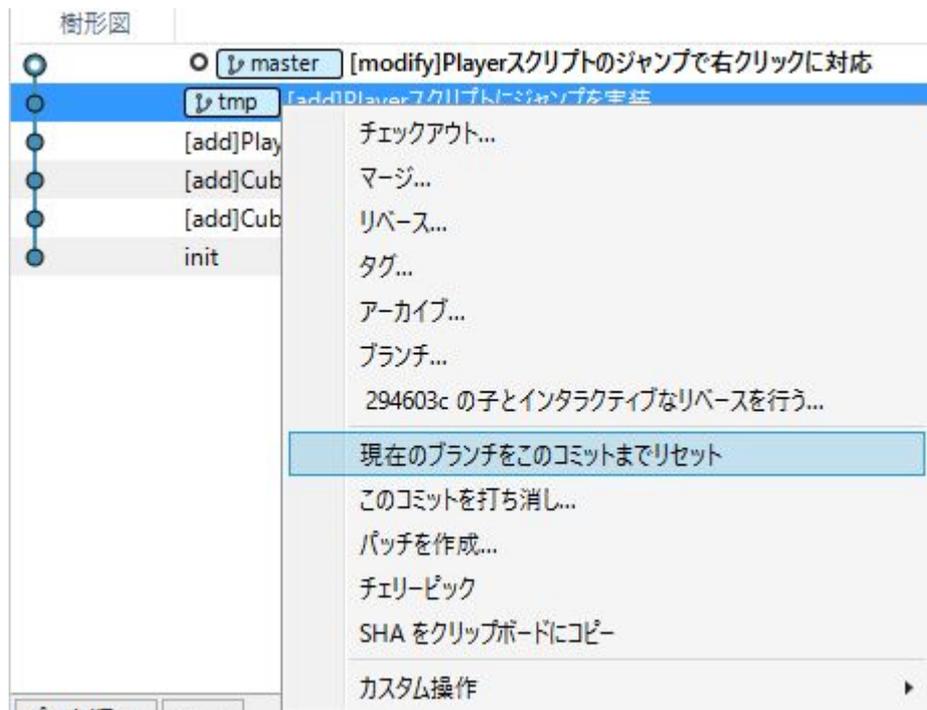
```
Assets/Player.cs  
21 21  
22 22     void Update()  
23 23     {  
24 24 -     _isJump = _isJump || Input.GetMouseButtonDown(0);  
24 24 +     _isJump = _isJump || Input.GetMouseButtonDown(0) || Input.GetMouseButtonDown(1);  
25 25     }  
26 26  
27 27     void FixedUpdate()
```

コミットを取消 (reset)

1. tmpを右クリックして、リセットを選択します。

バックアップしていた

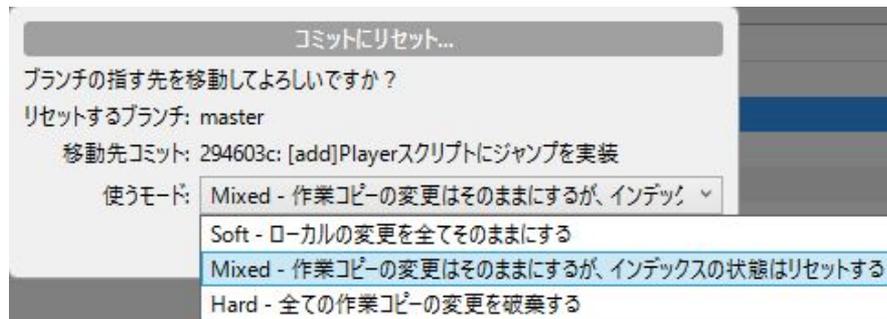
tmpブランチの状態に戻り、「右クリックに対応」のコミットは取消されます。



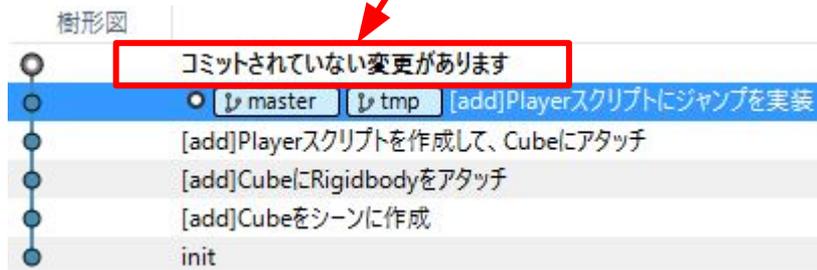
コミットを取消 (reset)

2. **Mixed**を選択して、リセットします。

Hardを選択すると「右クリックに対応」のコードが消えてしまいます。

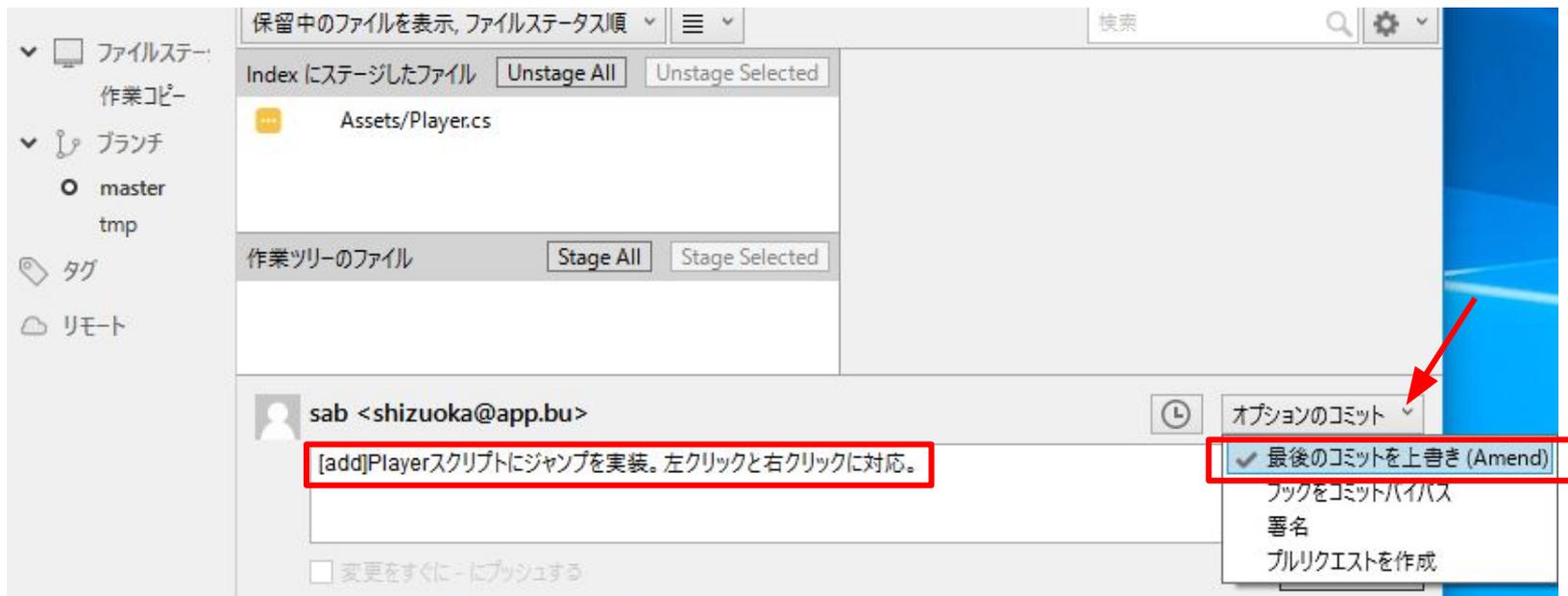


コミットする前の状態に戻ります。
「右クリックに対応」のコードは
残ります。



直前のコミットを修正 (*amend*)

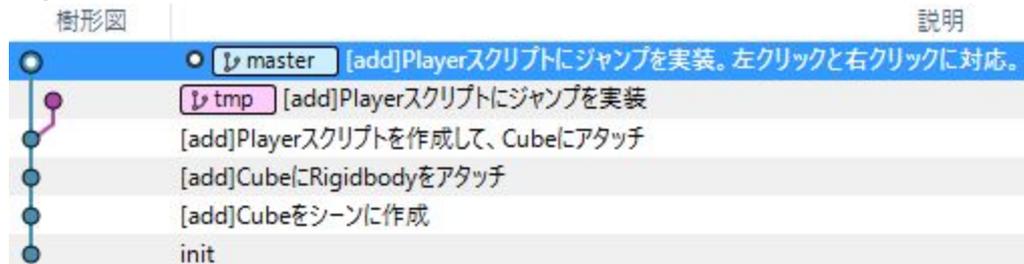
1. 最後のコミットを上書きにチェックを入れます。
2. メッセージを修正して、コミットします。



直前のコミットを修正 (*amend*)

3. 修正したコミットが作成されます。

mastert と tmpのコミット内容を見比べてみましょう。



この2つのコミットは**完全に別物**です。リビジョンが異なります。

```
コミット: b263fb9934fc8ded72c9c96bf226c8392de48d04 [b263fb9]
```

```
親: 7a050598ee
```

```
作者: sab <shizuoka@app.bu>
```

```
日時: 2016年6月2日 22:33:38
```

```
コミットした日時: 2016年7月30日 6:17:16
```

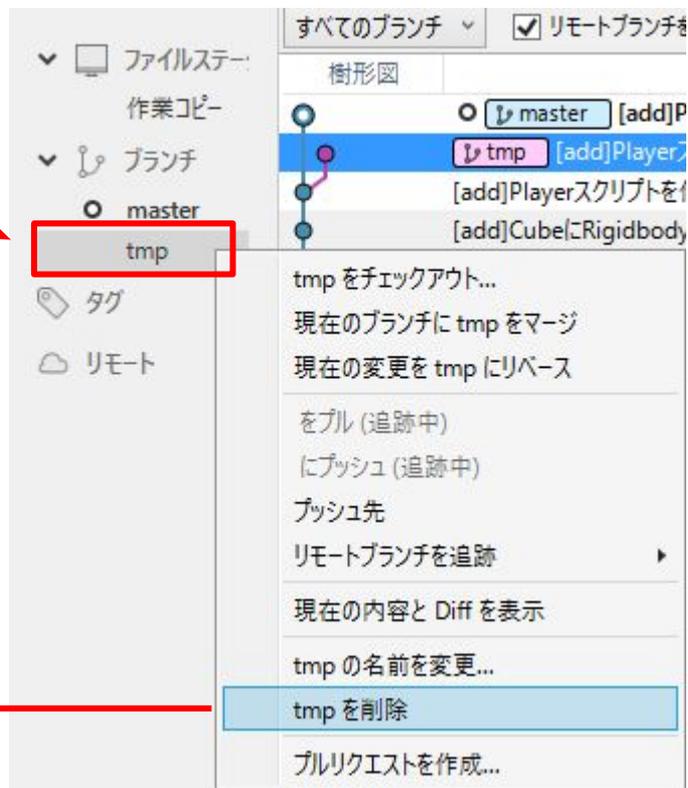
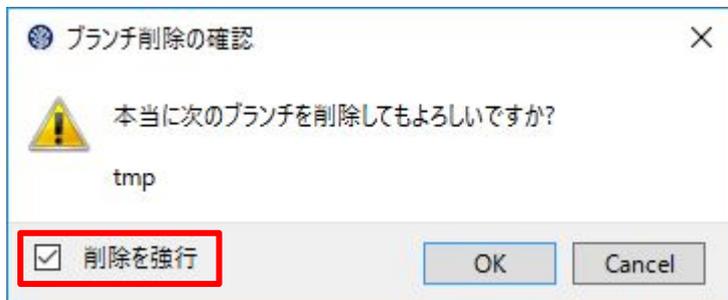
```
ラベル: HEAD, ->, master
```

```
[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。
```

直前のコミットを修正 (*amend*)

- tmp ブランチを右クリックして削除します。

コミットの修正が完了したので
バックアップ目的の
tmp ブランチは不要です。



コミット修正が完了

「右クリックに対応」のコミット修正が完了しました。



補足

- **ブランチ (*branch*)**
ブランチ名を“tmp/1”のように“/”で区切ると、ツリー表示されます。
- **リセット (*reset*)**
リセットはコミットを取消して過去に戻るだけでなく、
任意のリビジョン(コミット)に移動できます。
- **直前のコミットを修正 (*amend*)**
2つ前以上のコミットを修正する場合は、
インタラクティブなリベースを行う必要があります。
慣れるまでは *amend* で修正したり、
新しくコミットして対応しましょう。

まとめ

- ブランチを作成 ([branch](#))
バックアップ目的で活用できます。
- コミットを取消 ([reset](#))
Hardでリセットすると、修正コードが消えます。
任意のコミットに移動できます。
- 直前のコミットを修正 ([amend](#))
修正後のコミットは別物になります。

第2章クリア

最強無敵のGitでコミットの修正ができました。
こまめにコミットとして、どんどん修正しましょう。

次回は、ハンク・変更の破棄を使って、
コミット内容を細かく調整していきましょう。