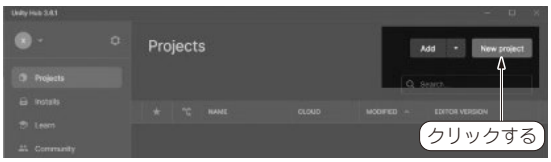
第2章 ゲームのロジックはPython→C#に変換して，色や配置の情報はパラメータとして設定する

ステップ②… Unityに移植する

山田 英伸



（a）プロジェクトの作成

（b）プロジェクトの準備

図1　Unityで新しいプロジェクトを用意する

プロジェクトやステージの作成

第1章で作成したPythonプログラムをゲーム・エンジンUnityで動くように移植します．Unityは個人利用であれば無償で使用できます（アカウントの作成は必要）．次のURLからダウンロードできます．

`https://unity.com/ja/download`

Unityを使う場合は，Pythonプログラミングのように全てをコードで準備することはできません．専用エディタでの作業が少し必要です．

● Unityで動くゲームは専用エディタで編集

Unityを起動するとUnityエディタと呼ばれる複数のビューからなるGUIアプリケーションが起動します．エディタの基本的な操作については，ある程度既知のものとして説明を続けます．不明なところがあれば，その操作についてもChatGPTに問い合わせして，操作のヒントをもらうこともできます．

このエディタ画面では，Inspectorと呼ばれるビューでパラメータを入力します．例えば，色や配置の情報です．

動作に関わるロジックはC#言語で実装します．つまりゲームのロジックは，第1章でPythonで作ったコードからC#に変換し，色や壁の配置情報といったものはパラメータとして設定します．

● ステップ1：プロジェクトの作成

Unity Hubを起動して，［New project］からプロジェクトを作成します．「2D（URP）Core」を選択します．「Connect to Unity Cloud」は未チェックの状態で進めます（**図1**）．

プロジェクトを開き，Projectビューで次の準備をします．

- Scenes以下にあるデフォルトのシーン名を「Game」に変更
- `Assets/Scripts`，`Assets/Prefabs`フォルダを作成

● ステップ2：ステージの作成

図2に示すようにGameObjectの階層構造を作成します．

次にWallsを右クリックして，「2D Object / Sprites / Square」を選択して，ステージの外壁や床を作ります（**図3**）．これらを作成後にはInspectorビューから「Add Component」ボタンを押して，「Physics 2D / Box Collider 2D」のコンポーネントを追加します．

● ステップ3：ブロックのPrefabを用意

UnityのPrefabとは，再利用を可能にするアセット（データ）の塊や一連のグループと考えることができます．Prefabは，再生成（インスタンス化）してScene上に複数配置が可能です．プログラム・コードからも制御できます．ここでは，丸いブロックの形状