

VSCode e Arduino IDE

Sumário

- **2. VSCode e Arduino IDE**
 - Instalando VSCode
 - Setando VSCode
 - Instalando a Arduino IDE
 - Setando a Arduino IDE

Instalando VSCode

1. **Acesse o site oficial:** <https://code.visualstudio.com/>
2. **Baixe o instalador** para o seu sistema operacional.
3. **Execute o instalador** e siga as instruções.
 - **Recomendação (Principalmente no Windows):** Durante a instalação, certifique-se de marcar as opções "Adicionar 'Abrir com o Code'..." (para pastas e arquivos) e, o mais importante, **"Adicionar ao PATH"**. Isso permitirá que você abra o VSCode facilmente pelo terminal com o comando `code ..`

Setando VSCode

Após instalar o VSCode, ele é um editor de texto genérico. Para transformá-lo em um ambiente de desenvolvimento C/C++ poderoso, precisamos instalar as extensões corretas.

Instalando Extensões Essenciais (C/C++)

As extensões fornecem recursos como autocompletar (IntelliSense), depuração, verificação de sintaxe e formatação de código.

1. Abra o Visual Studio Code.
2. Clique no ícone de **Extensões** na barra lateral esquerda (ou pressione `Ctrl+Shift+X`).
3. Na barra de pesquisa, digite `C/C++`.
4. Procure e instale o **C/C++ Extension Pack** (ID: `ms-vscode.cpptools-extension-pack`).

Este é o pacote oficial da Microsoft e é a melhor escolha, pois ele instala automaticamente um conjunto de ferramentas úteis, incluindo:

- **C/C++ (ID: `ms-vscode.cpptools`):** Esta é a extensão principal. Ela fornece o IntelliSense (autocompletar inteligente), navegação de código e suporte à depuração.
- **C/C++ Themes:** Temas de cores otimizados para código C/C++.
- **CMake Tools:** Suporte para projetos que usam o sistema de build CMake.

Verificando a Conexão com o Compilador

Com as extensões instaladas, o VSCode agora precisa "enxergar" os compiladores (`gcc` e `g++`) que você instalou anteriormente.

Felizmente, se você seguiu os passos anteriores e **adicionou o bin do seu compilador (seja do MSYS2, build-essential ou Homebrew) ao PATH do sistema**, a extensão C/C++ da Microsoft geralmente o detectará automaticamente.

Você saberá que funcionou quando abrir um arquivo `.c` ou `.cpp` e o IntelliSense (autocompletar de código) começar a funcionar corretamente.

⚡ Instalando a Arduino IDE

A **Arduino IDE** (Ambiente de Desenvolvimento Integrado) é o software oficial da Arduino. É onde você vai escrever seu código em C/C++, compilá-lo e enviá-lo (fazer upload) para a sua placa Arduino.

A versão moderna (recomendada) é a 2.0 ou superior, que é mais rápida e tem recursos modernos como autocompletar.

1. **Acesse o site oficial:** Vá para a página de software da Arduino: <https://www.arduino.cc/en/software>
2. **Baixe o Instalador:**
 - **Windows:** Baixe o "**Windows Win 10 and newer, 64 bits**". É um instalador (`.exe`) que cuidará de tudo para você, incluindo a instalação de drivers básicos.
 - **macOS:** Baixe o arquivo `.dmg` para o seu processador (Apple Silicon ou Intel).
 - **Linux:** Você pode baixar o `.AppImage` ou instalar via gerenciador de pacotes.
3. **Instale o Software:** Execute o instalador que você baixou. Siga as instruções na tela. Durante a instalação (principalmente no Windows), ele pode perguntar se você deseja instalar "drivers" ou "portas seriais (COM)". **Aceite e instale todos** eles, pois são essenciais para que o computador possa se comunicar com as placas.

⚙️ Setando a Arduino IDE

Ao abrir a IDE pela primeira vez, ela já está pronta para as placas mais comuns, como o **Arduino Uno**. Vamos apenas ajustar algumas configurações importantes para organizar seu trabalho.

1. 📁 Configurando o Local do Sketchbook

Este é um passo muito importante para manter seus projetos organizados. O "Sketchbook" é a pasta principal onde a Arduino IDE salva todos os seus projetos (sketches) e onde ela procura por bibliotecas que você instala manualmente.

Por padrão, ela fica em `Documentos/Arduino`. É uma boa ideia mudar isso para uma pasta de projetos dedicada (talvez uma pasta sincronizada com a nuvem, como Google Drive ou Dropbox).

1. Abra a Arduino IDE.
2. Vá em **Arquivo > Preferências** (No macOS, é **Arduino IDE > Settings...**).
3. Encontre o campo chamado "**Localização do Sketchbook**".
4. Clique em "**Procurar**" (Browse) e escolha a pasta onde você quer que seus projetos de Arduino fiquem.
5. Clique em **OK**. A IDE talvez precise ser reiniciada.

2. 📖 Instalando Bibliotecas (Libraries)

Bibliotecas (ou "libs") são pacotes de código prontos que facilitam tarefas complexas (ex: controlar um display, um sensor específico, etc.).

- **Método 1: Gerenciador de Bibliotecas (Recomendado)**

1. Clique no ícone de Livros 📖 na barra lateral esquerda (Gerenciador de Bibliotecas).
2. Na barra de pesquisa, digite o nome da biblioteca que você precisa (ex: **Adafruit GFX**).
3. Encontre-a na lista e clique em **Install** (Instalar).

- **Método 2: Instalar por .zip**

1. Se você baixou uma biblioteca de um site (como o GitHub) em um arquivo **.zip**.
2. Na IDE, vá em **Sketch > Incluir Biblioteca > Adicionar .ZIP Biblioteca...**
3. Selecione o arquivo **.zip** que você baixou. (A IDE vai copiá-la automaticamente para a sua pasta de Sketchbook).

3. ✅ Testando a Configuração (O "Blink")

Este é o "Hello, World!" do hardware. Vamos garantir que seu PC consegue conversar com seu Arduino.

1. **Conecte sua placa** (Arduino Uno) ao computador com um cabo USB.

2. **Selecione a Placa:**

- A IDE 2.0 é inteligente e geralmente detecta a placa automaticamente. Você deve vê-la no menu suspenso no topo (ex: "Arduino Uno em COM3").
- Se não, clique nesse menu ou vá em **Ferramentas > Placa** e procure sua placa (ex: **"Arduino Nano"**).

3. **Selecione a Porta:**

- A porta (a "porta USB" lógica) também deve ser detectada automaticamente.
- Se não, vá em **Ferramentas > Porta**.
- Você verá uma porta COM (no Windows, ex: **COM3**) ou **tty** (no Linux/Mac). Selecione-a. (Se você não sabe qual é, desconecte a placa, veja qual porta sumiu, e conecte-a novamente).

4. **Abra o Exemplo:**

- Vá em **Arquivo > Exemplos > 01.Basics > Blink**.

5. **Faça o Upload:**

- Clique no ícone de **seta para a direita (→)** na parte superior para compilar e enviar o código para a placa.
- Você verá a barra de progresso. Se tudo der certo, o LED "L" na sua placa começará a piscar!

4. 💡 Outras Dicas de Configuração (Recomendado)

Ainda na tela de **Arquivo > Preferências**, habilite estas duas opções. Elas ajudam **muito** na depuração:

- ☒ **Mostrar números de linha:** Essencial para encontrar erros que o compilador aponta.
- ☒ **Mostrar saída detalhada durante: [Compilação]**
 - Isso é **extremamente útil**. Quando seu código falhar ao compilar, a janela de saída preta na parte inferior mostrará *exatamente* qual comando falhou e o erro completo, em vez de apenas uma mensagem genérica.