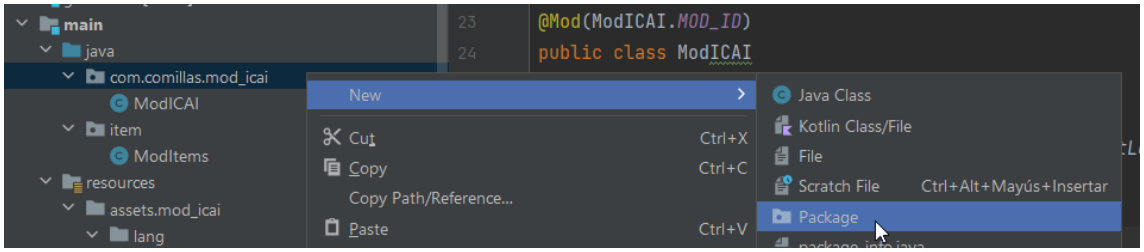
Práctica 5. Creación de bloques

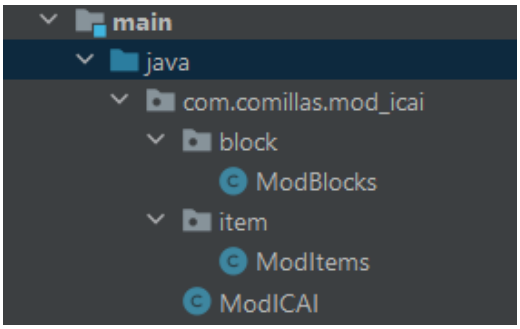
Introducción

En este documento se explicará qué pasos se deben seguir para crear un bloque personalizado de Comillas. Dicho bloque podrá colocarse en el mundo de Minecraft, como cualquiera del resto del juego.

Instrucciones

Comenzaremos creando la clase ModBlocks en el nuevo directorio creado **block**.





Este sería el código que compone la clase:

```
package com.comillas.mod_icai.block;

import com.comillas.mod_icai.ModICAI;
import com.comillas.mod_icai.item.ModItems;
import net.minecraft.world.item.BlockItem;
import net.minecraft.world.item.CreativeModeTab;
import net.minecraft.world.item.Item;
import net.minecraft.world.level.block.Block;
import net.minecraft.world.level.block.state.BlockBehaviour;
import net.minecraft.world.level.material.Material;
import net.minecraftforge.eventbus.api.IEventBus;
import net.minecraftforge.fmllegacy.RegistryObject;
import net.minecraftforge.registries.DeferredRegister;
import net.minecraftforge.registries.ForgeRegistries;

import java.util.function.Supplier;

public class ModBlocks {
    public static final DeferredRegister<Block> BLOCKS =
            DeferredRegister.create(ForgeRegistries.BLOCKS,
ModICAI.MOD_ID);

    public static final RegistryObject<Block> COMILLAS_BLOCK =
registerBlock("comillas_block",
            ()-> new
Block(BlockBehaviour.Properties.of(Material.METAL).strength(10f)));


    private static <T extends Block>RegistryObject<T>
registerBlock(String name, Supplier<T> block){
        RegistryObject<T> toReturn = BLOCKS.register(name,block);
        registerBlockItem(name, toReturn);
        return toReturn;
    }

    private static <T extends Block> void registerBlockItem(String
name, RegistryObject<T> block){
        ModItems.ITEMS.register(name, () -> new
BlockItem(block.get(),
                new
Item.Properties().tab(CreativeModeTab.TAB_MISC)));
    }

    public static void register(IEventBus eventBus){
        BLOCKS.register(eventBus);
    }
}
```

Funcionalidad del código que acabamos de copiar:

- El atributo de clase BLOCKS será un DeferredRegister: un tipo de clase encargado de que Forge pueda registrar nuestros bloques.
- Para crear nuestro bloque personalizado, añadiremos el atributo de clase COMILLAS_BLOCK, que es un RegistryObject, que asignaremos mediante el método registerBlock(String name, Supplier<T> block), con los dos siguientes argumentos:
    - Name: que será el String que identifique a nuestro bloque (“comillas_block”)
    - Block: que implementaremos mediante la siguiente expresión lambda:

      ()-> new Block(BlockBehaviour.Properties.of(Material.METAL).strength(10f)));

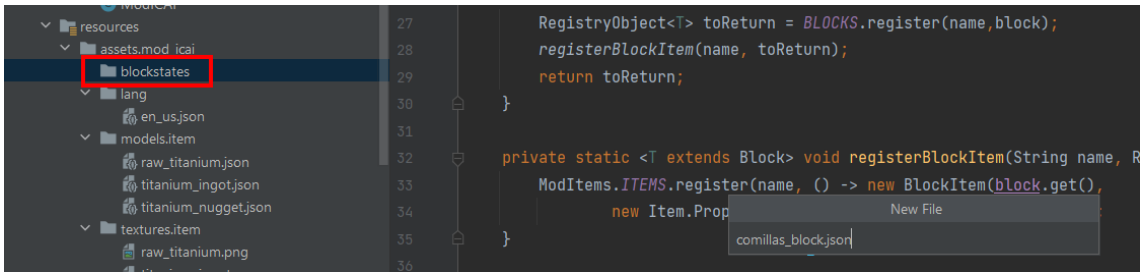
      Tal y como se puede observar, esta expresión lambda invoca el constructor de la clase Block, de manera que podemos asignarle propiedades a nuestro bloque.
- El método *registerBlock*, mencionado anteriormente, genera el RegistryObject mencionado anteriormente, y llama al método registerBlockItem.
- El método *registerBlockItem* se encarga de registrar nuestro bloque a modo de Item, de manera análoga a la creación de Items vistas en otros documentos.
- Por último, el método *register* registra BLOCKS en el bus de eventos.

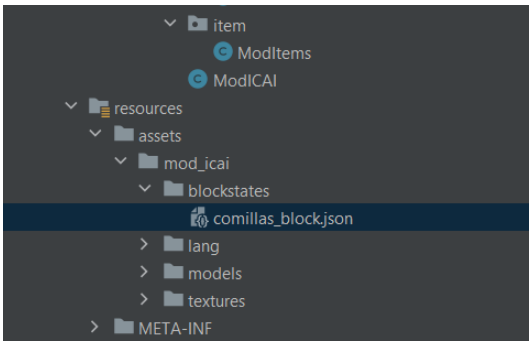
A continuación, tenemos que mandar al método *register* desde nuestra clase **ModICAI**:

```
public ModICAI() {
    // Register the setup method for modloading
    IEventBus eventBus = FMLJavaModLoadingContext.get().getMod
    eventBus.addListener(this::setup);

    ModItems.register(eventBus);
    ModBlocks.register(eventBus);

    // Register ourselves for server and other game events we
    MinecraftForge.EVENT_BUS.register( target: this);
}

private void setup(final FMLCommonSetupEvent event)
{
    // some preinit code
    LOGGER.info("HELLO FROM PREINIT");
    LOGGER.info( message: "DIRT BLOCK >> {}", Blocks.DIRT.getReg
```

Por último, debemos crear varios archivos *json* con información sobre nuestro bloque:

Comenzaremos con un json con el nombre que le hayamos puesto a nuestro bloque cuando hicimos la llamada a registerBlock ("comillas_block"):

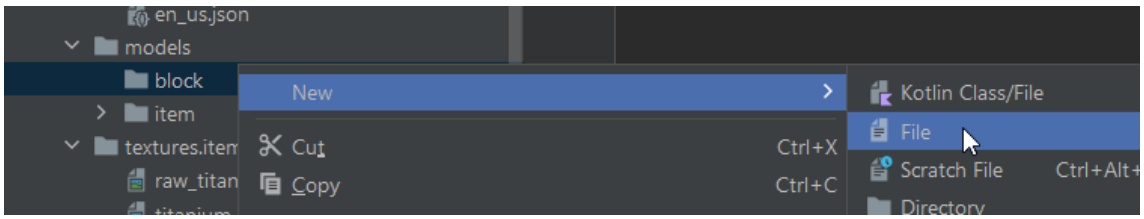


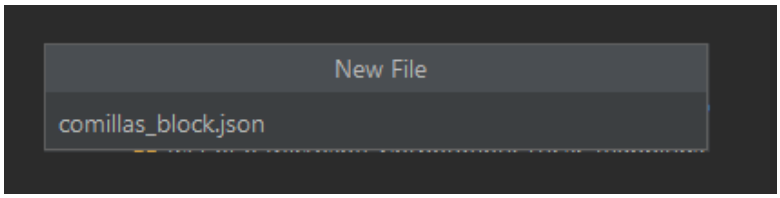


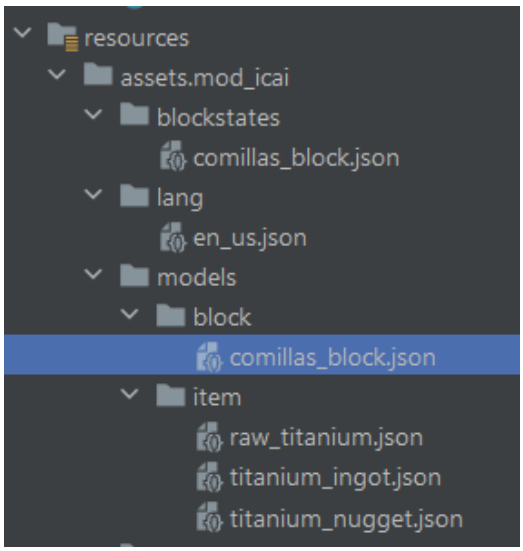

```
{
  "variants": {
    "": {"model":  "mod_icai:block/comillas_block"}
  }
}
```

En este *json*, especificaremos qué posibles estados tiene nuestro bloque (en nuestro caso solamente el estándar). Se debe tener especial cuidado a la hora de escribir correctamente nuestro modid.

Crear un fichero comillas_block.json que especifique las texturas a manejar el bloque dentro del directorio models\block.





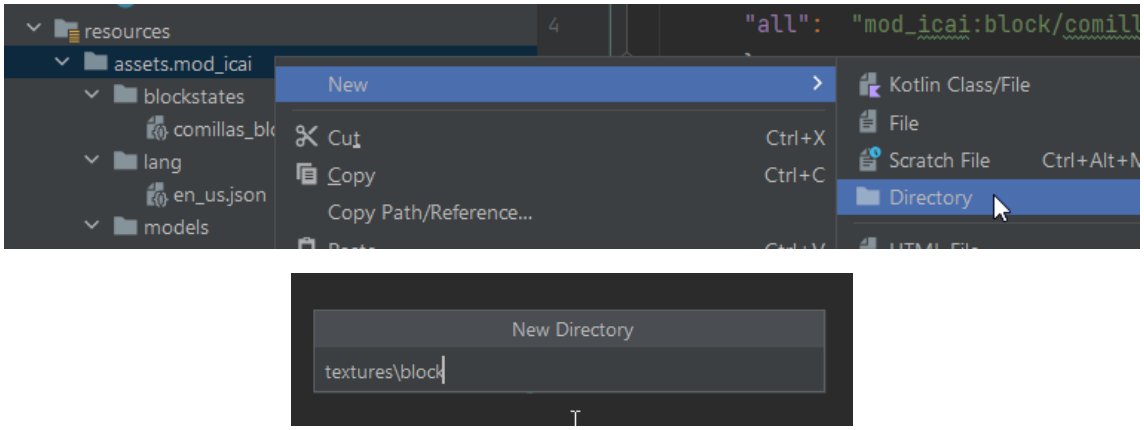


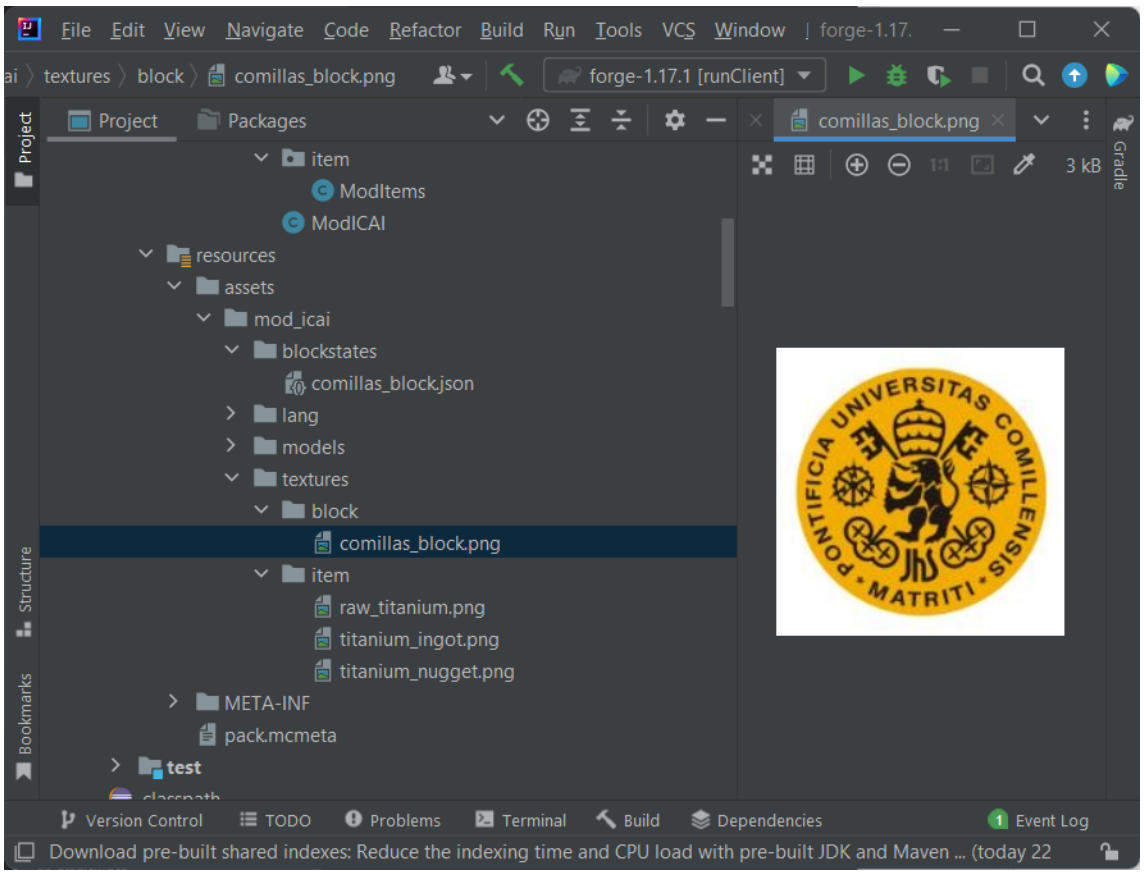
En este json especificamos qué texturas debería tener nuestro bloque. En este caso solamente será una:

```
{
  "parent": "block/cube_all",
  "textures": {
        "all":  "mod_icai:block/comillas_block"
        }
}
```

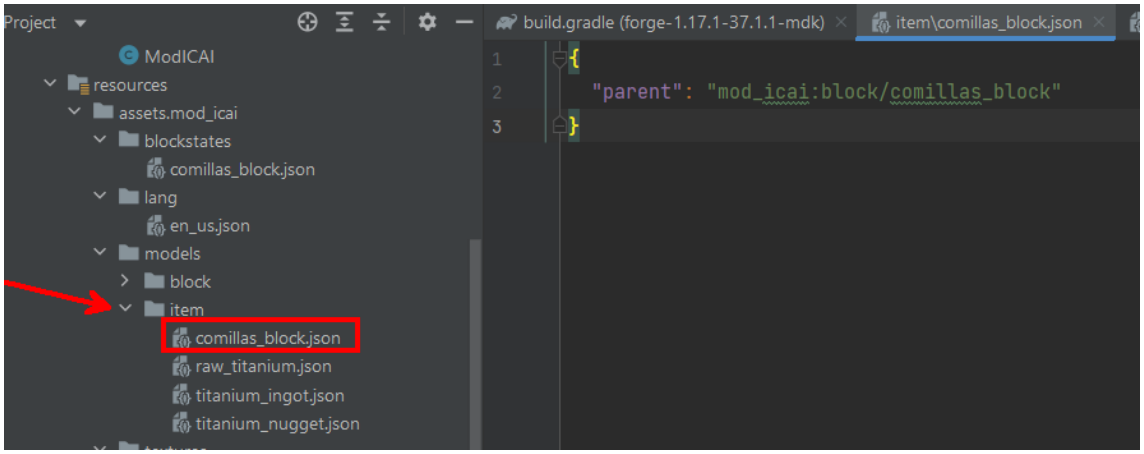
Lo que estamos indicando es que en el directorio textures/block nuestra textura estará definida por la imagen comillas_block.png

Ahora nos faltaría crear el directorio de texturas de bloques y guardar la imagen en él:

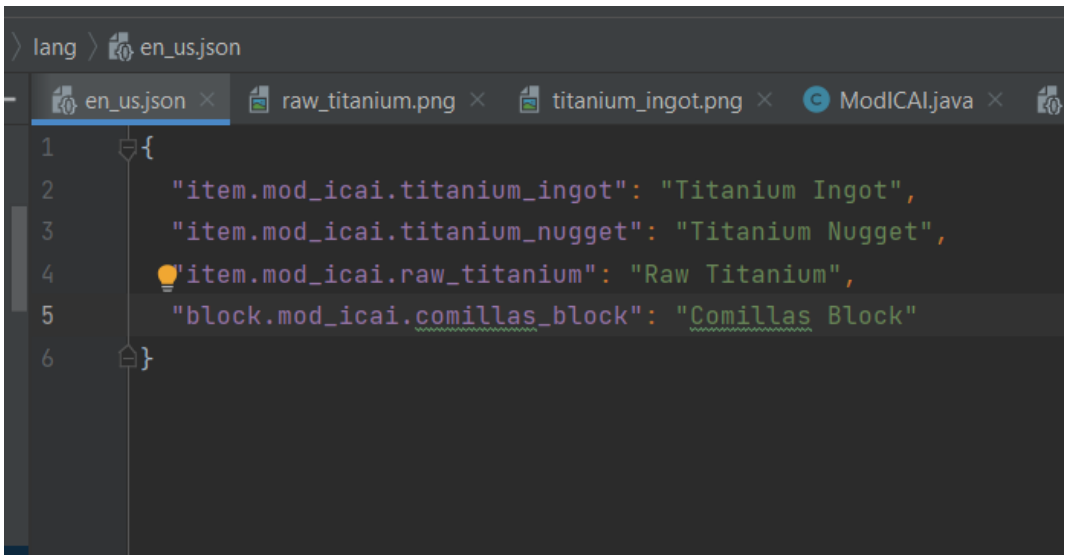




A continuación, necesitaríamos otro fichero **comillas_block.json** que simplemente servirá para indicar que nuestro bloque estará asociado a un item:



Por último, debemos indicar el nombre que aparecerá en Minecraft para identificar nuestro bloque. Esto se especificaba en el archivo en_us.json:



Si ejecutamos Minecraft, podremos comprobar cómo ya contamos con nuestro bloque disponible. Lo podemos arrastrar hasta nuestra mochila de ítems favoritos.