**GitHub**

GitHub je online platforma pro správu verzí a spolupráci na softwarových projektech, která využívá systém Git. Umožňuje vývojářům spravovat svůj kód, sledovat změny a efektivně spolupracovat v týmech.¨

GitHub je zásadní nástroj pro týmovou spolupráci a open-source vývoj, který usnadňuje sdílení a správu kódu na globální úrovni.

**Klíčové pojmy:**

1. **Pull** **from** **GitHub** – Stáhnutí nejnovějších změn z repozitáře na GitHubu do lokálního počítače. Provádí se příkazem **git pull**, který aktualizuje lokální kód na základě vzdáleného repozitáře.
2. **Push** **to** **GitHub** – Odeslání lokálních změn do vzdáleného repozitáře na GitHubu. Používá se příkaz **git push**, který přenese potvrzené změny (commity) do hlavní nebo jiné větve projektu.
3. **GitHub Fork** – Vytvoření kopie existujícího repozitáře do vlastního GitHub účtu. Fork umožňuje nezávislou práci na projektu bez ovlivnění původního repozitáře.
4. **GitHub Send Pull Request** – Žádost o začlenění změn z jedné větve nebo forku do hlavního repozitáře. Vývojář odešle pull request, který umožňuje správci projektu zkontrolovat a schválit změny před jejich začleněním.
5. **GitHub** **Sync** – Synchronizace lokálního repozitáře se vzdáleným repozitářem na GitHubu. Pomáhá udržet lokální verzi aktuální vůči změnám provedeným jinými vývojáři. Obvykle se provádí kombinací příkazů **git fetch** a **git merge** nebo jednoduše **git pull**.
   1. **Git** **Fetch** – Příkaz git fetch stáhne nejnovější změny ze vzdáleného repozitáře, ale neaplikuje je automaticky na lokální větev. Umožňuje vývojářům prohlédnout změny před jejich začleněním pomocí git merge nebo git rebase.
   2. **Git** **Rebase** – Příkaz git rebase umožňuje přesunout (přepsat) změny z jedné větve na druhou tak, aby se historie změn udržela lineární. Na rozdíl od git merge, který uchovává historii více větví, git rebase přehrává commity z jedné větve na druhou, což vede k čistší a přehlednější historii repozitáře.

**GitHub Branch** – Větev (branch) umožňuje paralelní vývoj bez ovlivnění hlavní větve (např. main nebo master). Pomocí **git** **branch** **<název\_větve>** se vytvoří nová větev a přepínání mezi větvemi se provádí pomocí git checkout nebo modernějšího git switch. Větve umožňují pracovat na nových funkcích, opravách chyb nebo experimentech, aniž by se měnila hlavní větev kódu.

**Mikrokontrolér** – Co to je? Mikrokontrolér (MCU) je integrovaný obvod obsahující procesor, paměť a periferní rozhraní na jednom čipu. Používá se pro řízení elektronických zařízení, automatizaci a vestavěné systémy. Oproti běžným procesorům je optimalizován pro nízkou spotřebu energie a specializované úlohy.

**ARM** - platforma ARM (Advanced RISC Machine) je architektura procesorů založená na principu RISC (Reduced Instruction Set Computing), což znamená, že používá jednoduché a efektivní instrukce. ARM procesory jsou široce využívány v mobilních telefonech, tabletech, mikrokontrolérech i průmyslových zařízeních díky jejich nízké spotřebě energie a vysokému výkonu.

**Příklady zařízení:**

1. **ESP** (např. ESP8266, ESP32) – WiFi a Bluetooth mikrokontroléry určené pro IoT aplikace.
2. **Arduino** – Otevřená platforma s různými modely mikrokontrolérů vhodná pro začátečníky i pokročilé projekty.
3. **Raspberry Pi (Rpi)** – Malý jednodeskový počítač s operačním systémem Linux, vhodný pro pokročilé aplikace a vývoj AI projektů.

**Možnosti programování a používané jazyky:**

1. C/C++ – Nejčastější volba pro nízkoúrovňové programování mikrokontrolérů.
2. Python – Používaný hlavně u Raspberry Pi, ale také v MicroPythonu a CircuitPythonu pro ESP a Arduino.
3. Arduino IDE (C++ s rozšířením) – Speciální prostředí pro Arduino mikrokontroléry s jednoduchou syntaxí.

**Způsoby komunikace:**

* **Sériová komunikace** (**UART**, **USART**) – Přenos dat pomocí dvou vodičů (Tx, Rx), např. pro debugování.
* **SPI** (Serial Peripheral Interface) – Rychlá synchronní komunikace pro připojení periferií, např. displejů nebo pamětí.
* **I2C** (Inter-Integrated Circuit) – Efektivní komunikace pro více zařízení po dvou vodičích (SDA, SCL).
* **SSH** (Secure Shell) – Síťová komunikace pro vzdálenou správu Raspberry Pi a dalších zařízení s Linuxem.
* **I/O piny** a jejich využití: I/O (Input/Output) piny umožňují komunikaci mezi mikrokontrolérem a vnějším světem. Slouží k připojení senzorů, LED diod, motorů a dalších periferií. Rozdělují se na:
* **Digitální** **piny** – Používají se pro spínání (HIGH/LOW stav).
* **Analogové** **piny** – Umožňují čtení analogových hodnot, např. z potenciometrů.
* **PWM** (Pulse Width Modulation) – Simulace analogového signálu pomocí digitálních výstupů.

**Porovnání napěťových úrovní signálů:**

* **ESP** (ESP8266/ESP32) – 3.3V logika.
* **Arduino** (Uno, Mega) – 5V logika (některé modely 3.3V).
* **Raspberry** **Pi** – 3.3V logika, vyšší napětí může poškodit GPIO.

**Porovnání platforem podle vhodnosti pro projekty:**

* ESP – Ideální pro IoT projekty, bezdrátovou komunikaci a senzory.
* Arduino – Skvělé pro začátečníky, jednoduché projekty a řízení periferií.
* Raspberry Pi – Nejvhodnější pro pokročilé aplikace, servery, zpracování obrazu a AI projekty.

Každá platforma má své výhody v různých typech projektů a volba závisí na konkrétním použití a požadavcích.

**Rozdíly v přípravě grafiky pro tisk a web/mobilní aplikace**

Při přípravě grafiky je důležité rozlišovat mezi tiskovými materiály a grafikou určenou pro web či mobilní aplikace. Tyto dvě oblasti mají zásadní rozdíly v použitých barevných modelech, rozlišení a práci s textem. Správná příprava grafiky zajistí kvalitní vizuální výstup a optimalizaci pro dané médium.

1. Barevné modely

* **Tisk** – Používá model CMYK (Cyan, Magenta, Yellow, Black), který je založen na subtraktivním míchání barev. Barvy se skládají ze čtyř základních tiskových barev a výsledný vzhled závisí na tiskové technologii a materiálu. Každý tiskový stroj může zobrazovat barvy mírně odlišně, proto se často používají kalibrované ICC profily pro dosažení přesných barevných výstupů.
* **Web** **a mobilní aplikace** – Používají model RGB (Red, Green, Blue), který je založen na aditivním míchání světla. RGB model poskytuje jasnější a sytější barvy, protože obrazovky vyzařují světlo. Při práci s digitální grafikou je důležité brát v úvahu různé typy obrazovek a kalibraci monitorů, které mohou barvy zobrazovat odlišně.

2. Rozlišení

* **Tisk** – Vyžaduje vysoké rozlišení pro ostrý výstup. Obvykle se pracuje s hodnotou 300 DPI (dots per inch), což zajišťuje dostatečnou kvalitu při tisku. V některých případech, například u velkoformátového tisku, může být rozlišení sníženo na 150 DPI, protože diváci budou obraz sledovat z větší vzdálenosti.
* **Web** **a mobilní aplikace** – Standardně se používá nižší rozlišení, nejčastěji 72–96 PPI (pixels per inch), protože vyšší rozlišení není na obrazovkách nutné a zbytečně by zvětšovalo soubory. U moderních zařízení s vysokým rozlišením (Retina displeje) se však využívají obrázky s vyšším PPI pro ostřejší zobrazení.

3. Práce s textem

* **Tisk** – Text musí být připraven ve vektorovém formátu, aby byl ostrý i při velkém zvětšení. Používají se fonty s důrazem na čitelnost a správné vykreslení při tisku. Doporučuje se převést text na křivky, aby se zabránilo problémům s chybějícími fonty při tisku.
* **Web** **a** **mobilní** **aplikace** – Text se zobrazuje pomocí systémových nebo webových fontů, které musí být optimalizovány pro různé velikosti obrazovek. Používají se technologie jako CSS font-face nebo Google Fonts. Při návrhu je třeba dbát na čitelnost textu na různých zařízeních a správnou kombinaci fontů pro zachování konzistence vizuálního stylu.

4. Formáty souborů

* **Tisk** – Nejčastěji se používají formáty PDF, EPS, AI nebo TIFF, protože podporují bezztrátovou kvalitu a práci s vektorovou grafikou. TIFF se často využívá pro bitmapové obrázky s vysokým rozlišením.
* **Web a mobilní aplikace** – Používají se komprimované formáty jako JPEG, PNG, GIF a SVG. JPEG je vhodný pro fotografie, PNG pro průhledné obrázky a SVG pro vektorovou grafiku, která zůstává ostrá při zvětšení.

Shrnutí Příprava grafiky pro tisk vyžaduje odlišný přístup než grafika pro digitální média. Klíčové rozdíly spočívají v barevném modelu (CMYK vs. RGB), rozlišení (300 DPI vs. 72 PPI), práci s textem (vektory vs. webové fonty) a používaných souborových formátech. Při navrhování je důležité brát tyto faktory v úvahu, aby byl výstup co nejkvalitnější pro danou platformu. Správně zvolené postupy pomáhají minimalizovat chyby a zajišťují optimální výsledek pro tiskové i digitální projekty.