Github

* Github: Host & review code, manage projects & build software
* Devops features: all size: feature, fixing bug, Collaborate on changes.
* Developers gather to make things better.

---

* Issues, notification, branches, commits, pull request
* webpage deploy (Git & Github), Fork vs clone.

Github

* Not only a platform but offers features to optimize.

Branching

* many ideas - some are ready & some are not
* create branch - Any change in branch doesn't affect main.

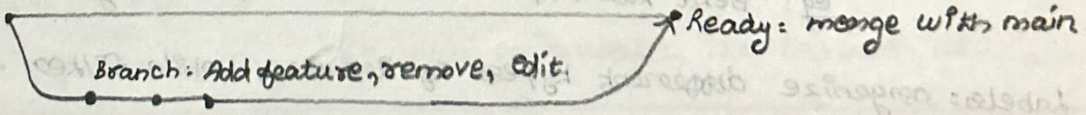
'Free to experiment'

'merge - when it is absolutely ready to go - (reviewed by Someone - Collaboratly)

Branching: core concept

* Anything in the main branch is always deployable.
* create branch off the main. (else affects main - no use of branching.

Branch name (descriptive): refactor - authentication
user - content - cache - key
make - retina - avators.



Github flow

Git: Distributed version control system - allows multiple devs to work on a project. (way to work with one/more local branches & pushes them to a remote repo).

* Git: Responsible for everything Github - related happens locally on your computer.

Key features of Git : * Branching
* version Control
* Installed & used on your local Computer.

Github : cloud platform - using Git as core tech. Simplifying the process of collaborating on project by
. * Providing website
* Command line tools
* Flow

"remote repository"

Git : Local computer

Github : uses Git : Sociopath.

Issues

* Communication b/w Consumer & dev team.
* An issue can be created - discuss topics, bugs, features, documentation classifications.
* Issues can assigned to owners, labels, projects & milestones.
* Associate issues with pull requests & other items to provide future traceability.

Milestones, Labels & Assignees

* must have edit access (issue) — Collaborator

Milestone : group of issues, corresponds to a project, feature or time.

eg : Beta Launch, October Sprint, Redesign

Labels : organize different types of issues : while filter - we can use label words - easy to view that particular issue.

Milestone :	like a folder - group
Labels :	Give name to identify.

Assignees

* "Responsible for moving that issue forward"
* Select the issue - menu will be there.

By using @ → we can notify other users.

Notifications: Any improvent/new things in issue - notify.

github.com/notifications

Refer: mastering markdown (styling)

mention: reference to other users (Twitter mentions)

Reference: 'one issue - dependent on other issue - related' ∴ we can give the reference.

Issue in another repo:

$$\begin{tikzcd} \text{kneath} \arrow[d] & \text{/example-project} \arrow[d] & \text{\#42} \arrow[d] \\ \text{Reponame} & \text{Issue} & \text{Number} \end{tikzcd}$$

---

* Preface: 'fixes', 'fix', 'closes', 'closed', 'close' → the commit is merged to main - automatically close the issue.

* Reference: easy in bug finding history tracking.

eg: fixed # 9196 - malformed HTML in doc.

Search: Keyword, State (closed), Assignee etc..

Issue dashboard

'Looking broader listing of all your issues - history'

github.com/issues

Pulse: shows everything happened in your repo.

eg: no. of commits, pull requests, new issues etc..

Branches

* 'preferred way to create changes in the Github flow'
* So that multiple people can work simultaneously - controlled way.
* Freedom to test - once ready - commit & merge to main via pull request.

---

pull request: signal - commits from one branch is ready to be merged into another branch.

often: dev request one or more reviewers to verify the code & approve the code. (Approved: Source request branch merged to base branch).

Labels: bug, docu, duplicate, help needed, improve, question

Actions

* Task automation - streamline processes in software development lifecycle.

workflow: Automated process added to repo.

Events : An activity triggers a workflow

Jobs : A set of steps that execute on a runner.

Steps : A task that can run one / more commands (actions)

Actions : Standalone commands - combined to steps (multiple-Job Steps)

Runners: server - having GitHub actions runner app installed.

eg: everytime a user creates a pull request.

Cloning & Forking

Cloning: Copy repo & its history to local machine. (Later we can push)

eg: git clone → command

gh repo clone

Forking

* make a copy of a repo in your account.
* Parent repo - upstream
* Forked copy - origin.

"once forked - clone - change then merge"

* without affecting upstream parent repo.
* If merge needed - pull request.

gitpages → hosting engine - right into your account.

Markup language

* Concise, lightweight syntax - Overhead inherent to HTML.

eg: Report a bug: depth & Context → * emphasize parts of your text
* List of reproduction steps
* Enumerate your observations
* Embed screenshots - links, logs.
* Reference other issues / lines of code.

* HTML - takes up lot of space - Markdown for help!

* Markdown - power of HTML & ease of plain text for editing.

Italics

* Use '*' or '_' b/w target text

This is *italic* text.
This is _italic_ text.

Bold

* Use ** or __

use * or _

* (or) _

*, _ → Italics   **, __ → bold , *, _ → Astrick & underscore.

Heading.

HTML → <h1> tag.

→ each for one level (1 to 6)

This is H6 text → 6th level heading.

Image

Image → ![Link an image](/learn/azure-devops/shared.png)

Link → [Link to learn](/learn)

List

* ordered lists start with numbers
* unordered - with asterisks or dashes.

1. First 1. Second 1. Third	- First - Nested - second - Third

Tables - 'use pipes'

pipes - Column (|)

dashes - Rows as a header (-)

- → designate previous row as a header.

eg:

```
First | second
 - | -
1| 2
3|4
```

First	second
1	2
3	4

o/p

Block quotes (>)

>This is quoted text

o/p

[ ] This is quoted text

Filling the gaps - 'we can use HTML also'
'when markdown - not supports'

Here is a <br /> line break

↳ line break.

Working with Code

This is 'code'

This is $\boxed{\text{code.}}$ → Rendered as code.

multiple lines

'''

→ Fenced code block.

GFM - offers syntax highlighting for popular languages! Just specify the language as part of the first tick sequence.

```
''' javascript
var first = 1;
var second = 2;
var sum = first + second;
'''
```

→ syntax highlighted.

Cross linking & pull requests

#Id → #3602

GH - 3602

desktop/desktop#3602

https://github.com/desktop/desktop/pull/3602

Linking specific commits

* URL
* SHA → its id
* User@SHA → username@Id
* username/repository@SHA.

Mention

@mention

Tracking task lists

- [x] First task
- [x] Second task
- [ ] Third task

$\xrightarrow{o/p}$

☑ First task

☑ Second task

☐ Third task.

---

* .md (or) .markdown extension
* Sharing snippets of text in Gists.

![ ] [link] → image

[Name] [link] → link

Unordered	Emojis	Lists (ordered)
* Item 1	:heart:	1. Item 1
* Item 2	:+1:	2. Item 2
*Item 2a	:smile:	1. Item 2a
*Item 2b	:sparkles:	2. Item 2b
	:tada:	

Github pages - Not hosting site (to host anything)

Settings → pages → activate pages

'static' - single page sites

YAML

Front matter - YAML metadata that prepends (adds at the beginning) the content of a file. It includes generator instructions to indicate the layout style of a markdown page (post, page..) - Title, page content variables, Author name.

```
---
layout : Post                                    → metadata
title  : This is set as the document tile.
---
This is visible body content      → HTML / md / Liquid templating.
```

Site is up & running:

* customize details about site via _config.yml
* _config.yml has metadata, navigation menus, themes, compiler options etc..

Github supports: HTML, md. (Jekyll supports Liquid template too)