Cheatography

RBA Git Commit Taxonomy Cheat Sheet

by Cash (CashM) via cheatography.com/123081/cs/48010/

Format

Pattern	Example
type(scope): summary	frame(program): redefine current problem state for AI-era redesign

Commit Types

Type	Use When	Typical Examples
`frame`	The governing understanding changes	problem reframing, scope clarification, correcting a higher-order assumption
`build`	Creating substantial new work	new artifacts, new sections, new matrices, new program/course content
`refactor`	Structure improves without major change to core intent	reorganizing documents, tightening structure, improving sequence or alignment
`synth`	Combining multiple artifacts into a higher-order output	merging sections into a full draft, combining several artifacts into one summary
`review`	A change is driven mainly by feedback or inspection	supervisor revisions, advisory revisions, review-response changes
`meta`	Process or repo guidance changes	workflow notes, commit taxonomy, README/process guidance
`chore`	Light maintenance only	renaming, moving files, formatting cleanup

Default Scopes

Scope	Use For
`program`	program design, transition models, bridge/successor work
`course`	course-level artifacts and redesign work
`whitepaper`	white paper sections, appendices, full draft work
`case-study`	case studies and reflections
`workflow`	process conventions, commit taxonomy, operational guidance
`repo`	repo-level organization and maintenance

Other scopes are allowed. Consistency is the key point.

Sample Commits

Situation	Sample Commit
redefine the core problem	`frame( pro gram): redefine bridge and successor relati onship`
add new course artifact work	`build( cou rse): add Python bridge implem ent ation patterns`
tighten existing program structure	`refact or( pro gram): tighten semest er- by- sem ester bridge structure`
merge paper sections into one draft	`synth( whi tep aper): merge sections and appendices into full draft`
revise after supervisor feedback	`review (wh ite paper): revise framing for supervisor readab ility`
update the commit system itself	`meta(w ork flow): simplify commit taxonomy for artifact ecosystems`
rename or reorganize files	`chore( repo): rename and reorganize support files`

Keep It Simple

Rule	Reminder

Keep It Simple (cont)

1	Prefer consistency over perfect precision
2	Use broad scopes, not overly specific ones
3	Avoid inventing new types unless needed repeatedly
4	If a commit spans multiple artifacts, classify by the dominant action

Quick Decision Rule

If the main move is ...	Use
changing the governing meaning	`frame`
creating substantial new material	`build`
reorganizing existing structure	`refactor`
combining multiple strands	`synth`
revising from feedback	`review`
updating process/repo guidance	`meta`
doing minor maintenance only	`chore`

RBA Definition

RBA *(Refraction-Based Architecture)* is a way of using AI to develop complex work through structured refraction rather than one-shot generation, so that meaning, architecture, and artifact coherence are preserved as the work evolves.
It is a governed human-AI collaborative process in which intent is progressively refracted across multiple artifact layers while preserving coherence, allowing lower-level work to inform and refactor higher-level understanding when reality reveals hidden constraints, opportunities, or structural mismatches.
(*Collaboration* was intentionally **deferred** because it semantically suggests two sentient beings working together, which risks anthropomorphizing AI in ways that are conceptually misleading. ***Refraction*** is more appropriate because, like light passing through a medium and changing direction or separating into visible components, it captures how human intent is perceived, transformed, decomposed, and delegated across different layers of artifacts, constraints, and AI-mediated processes.)