GIT - Das Wichtigste auf einer Seite

Was?	Befehl für Kommandozeile	Befehl für TortoiseGIT
Start:		
Neues Repository im aktuellen Ordner	git init	„Git Create repository here...”
Auf Server: nicht ausgechecktes GIT	git init --bare	
Mit vorhandenem Repo arbeiten	git clone <Adresse>	„Git Clone ...“ Adresse und Ordner angeben
Globale Einstellungen setzen	git config --global user.email <EMail>, git config --global user.name <Name>	Name und EMail unter „Settings”→ „Git”→ „Config”→ „Save as Global“ eingeben
Neue Datei ins GIT hinzufügen	git add <myFile>	„Add ...”
Commit	git commit -a -m für Commitmessage	„Working dir changes“ → Rechtsklick „Commit ...”
Status abfragen	git status	Icons zeigen Status von Dateien und Ordnern an
Unterschiede anzeigen	git diff [file]	Rechtsklick „Compare with Base“ oder „Show differences as unified diff”
Synchronisation mit dem Server:		
Änderungen vom Server holen	git fetch	„Fetch...”
Änderungen vom Server holen + Auto-Merge	git pull	„Pull...”
Änderungen an Server schicken	git push	”Push...”
Branches:		
Einen neuen Branch erstellen	git branch <newBranch> master	„Create Branch at this version...”
In einen Branch wechseln	git checkout <otherBranch>	„Switch/Checkout to this...”
Neuen Branch erstellen und direkt wechseln	git checkout -b <newBranch>	
Branch für alle Verfügbar machen	git push origin <localBranch>	Branch beim „Push“, als „Local:“ sowie als „Remote:“ eintragen
Merging = Konfliktbehandlung:		
Komplett die Änderungen der anderen Version übernehmen	git checkout	„Resolve conflict using theirs”
Komplett die Änderungen des eigenen Commit übernehmen	git merge -s ours	„Resolve conflict using mine”
Branches in den Master einpflegen	git merge	Verwendung des „Mergetools“
Parameter für git merge:		
Nur von konfliktbehafteten Stellen unsere Version verwenden	git merge -s recursive -Xours	
Nur von konfliktbehafteten Stellen die andere Version verwenden	git merge -s recursive -Xtheirs	
Änderungen die nur Leerzeichen betreffen ignorieren	git merge -s recursive -Xignore-space-change / -Xignore-all-space / -Xignore-space-at-eol	
Gleicher Algorithmus wie merge ohne parameter, aber nicht rekursiv	git merge -s resolve	
Mergen mehrerer Branches in einen Branch, führt keine komplexe Konfliktbehandlung durch	git merge -s octopus	
Weiteres:		
Änderungen rückgängig machen	git checkout HEAD <myFile>	„Revert...”
Ganzen Commits rückgängig machen (inkl. Versionen nach dem Commit)	git reset --hard HEAD	„Reset master to this...”
Nur einen bestimmten Commit rückgängig machen	git revert <Nummer>	„Revert change by this commit”
History anzeigen	git log / grafisch: gitk, qgit	„Show log”
Tags setzen	git tag -a newTag <tag>	„Create Tag at this version...”
Tags pushen	git push --tags	beim „Push...“ „Include Tags“ setzen