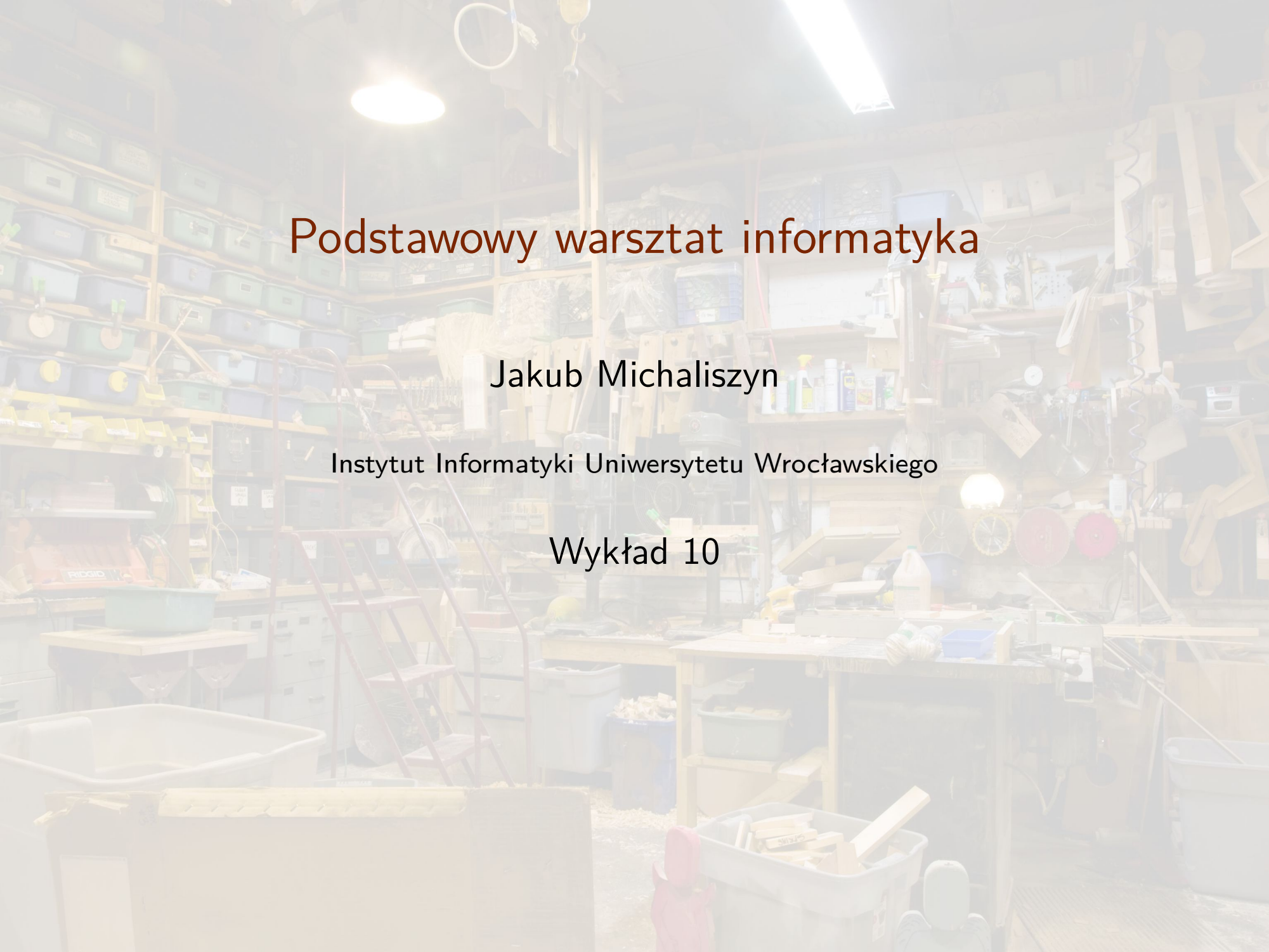


Podstawowy warsztat informatyka

Jakub Michaliszyn

Instytut Informatyki Uniwersytetu Wrocławskiego

Wykład 10

Przypomnienie – forki i pull requesty

Problem: chcemy współpracować z różnymi ludźmi. Chcemy, aby owoce ich pracy lądowały w naszym repozytorium, ale żeby nie mogli nic popsuć.

Przypomnienie – forki i pull requesty

Problem: chcemy współpracować z różnymi ludźmi. Chcemy, aby owoce ich pracy lądowały w naszym repozytorium, ale żeby nie mogli nic popsuć.

Rozwiązanie: współpracownicy tworzą kopię repozytorium (forka), a po skończonej pracy proszą o to, byśmy wciągnęli ich kod do naszego repozytorium (pull request).

Problem 1

Problem: Chcemy odrzucić lokalne, nieskomitowane zmiany.

Problem 1

Problem: Chcemy odrzucić lokalne, nieskomitowane zmiany.

Popularne rozwiązanie (niezbyt dobre?) `git stash`

Problem 1

Problem: Chcemy odrzucić lokalne, nieskomitowane zmiany.

Popularne rozwiązanie (niezbyt dobre?) `git stash`

git stash odkłada zmiany na później, nie kasuje ich.

Problem 1

Problem: Chcemy odrzucić lokalne, nieskomitowane zmiany.

Popularne rozwiązanie (niezbyt dobre?) `git stash`

git stash odkłada zmiany na później, nie kasuje ich.

Rozwiązanie: git checkout – pliki

Problem 2

Problem: chcemy usunąć/przenieść pliki tak, żeby git wiedział o tej operacji.

Problem 2

Problem: chcemy usunąć/przenieść pliki tak, żeby git wiedział o tej operacji.

git rm/mv
git mv plik1 plik 2 wykonuje:

- mv plik1 plik2
- git remove plik1
- git add plik2

Przy przenoszeniu plików należy zachować szczególną ostrożność!

Problem 2

Problem: chcemy usunąć/przenieść pliki tak, żeby git wiedział o tej operacji.

git rm/mv
git mv plik1 plik 2 wykonuje:

- mv plik1 plik2
- git remove plik1
- git add plik2

Przy przenoszeniu plików należy zachować szczególną ostrożność!

git reset plik – usuwa z gita, ale nie z komputera

Problem 3

Problem: chcemy usunąć lokalny commit.

Problem 3

Problem: chcemy usunąć lokalny commit.

`git reset HEAD~3` zachowuje lokalne zmiany
`git reset --hard HEAD~3` usuwa lokalne zmiany (wszystkie!).

Problem 4

Problem: chcemy usunąć zdalny commit.

To jest prawie zawsze zły pomysł. Chyba, że pracujemy sami.

Problem 4

Problem: chcemy usunąć zdalny commit.
To jest prawie zawsze zły pomysł. Chyba, że pracujemy sami.

git revert HEAD
git revert -n HEAD – bez dodatkowego commita

Problem 5

Problem: chcemy ustrzec się przed przypadkowym dodaniem niektórych plików, np. baza.db, config.php itd.
Rozwiązanie: dodać pliki go .gitignore. Przykładowy plik:

```
*.dll
*.exe
*.o
*.so
*.7z
*.log
*.sql
*.sqlite
.DS_Store?
.Trashes
ehthumbs.db
Thumbs.db
```

Problem 6

Problem: coś się popsuło, ale nie wiadomo kiedy.

Problem 6

Problem: coś się popsuło, ale nie wiadomo kiedy.

Rozwiązanie: przeszukiwanie binarne.

```
git bisect start          # rozpoczęcie procesu
git bisect bad            # oznaczamy obecną wersję jako złą
git bisect good revision # oraz oznaczamy ostatnią
                            znaną dobrą wersję
```

Potem powtarzac do rozwiązania:

```
git bisect good lub git bisec bad
```

git hooks

Hooks - skrypty automatyzujące pracę. Mogą być lokalne i zdalne.

git hooks

Najbardziej przydatne lokalne hooki

- pre-commit
- prepare-commit-msg
- commit-msg
- post-commit
- post-checkout
- pre-rebase

pre-commit

Wywołuje się, gdy chcemy stworzyć komit.
Jeśli zwraca 0, można stworzyć komit, inaczej jest to zabronione.
Można wymusić komit dopisując opcję –no-verify.
Przykład: zabroń komita, jeśli są zmodyfikowane pliki niedodane do komita:

```
#!/bin/sh
if [ ! -z \
  "`git status | grep 'Changes not staged for commit'`"]
then
        echo "Zapomniałeś czegoś dodać."
        exit 1
else
        exit 0
fi
```

prepare-commit-msg

Wejście zawiera nazwę pliku z treścią opisu komita.

prepare-commit-msg

Wejście zawiera nazwę pliku z treścią opisu komita.

```
#!/bin/bash

branchName=‘git branch | grep ’^\*’ | cut -b3-‘

firstLine=‘head -n1 $1‘

if [ -z "$firstLine" ] ;then # jeśli to nie jest amend
    sed -i "1s/^/$branchName:␣\n/" $1
fi
```

commit-msg

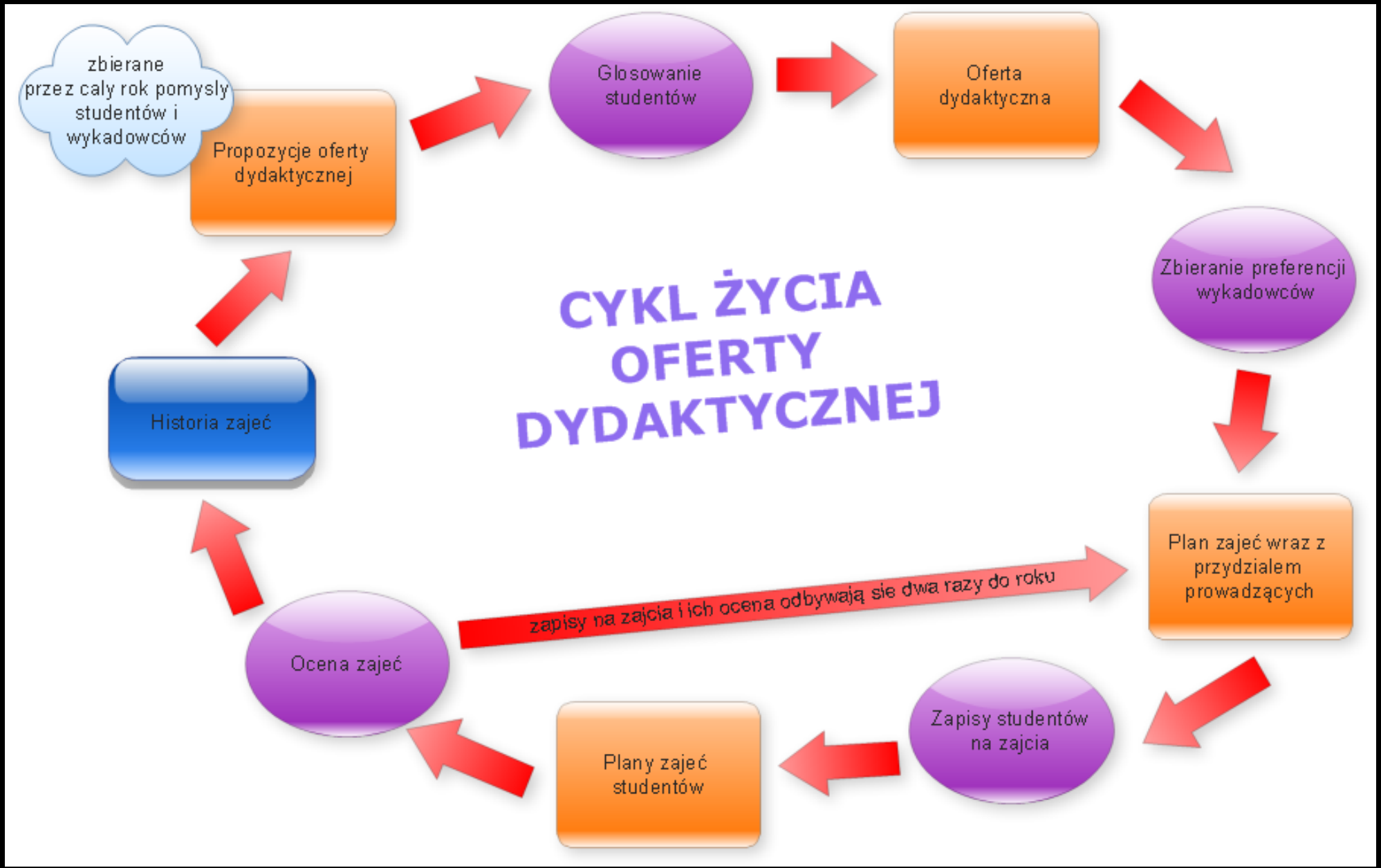
```
#!/bin/sh
test="`grep ␣'Kurde'␣$1␣`"
if [ ! -z "$test" ]
then
        echo "Coś␣się␣stało?"
        read line
        echo "Zaiste␣ambaras.␣Przygotuję␣ci␣melisę"
        ~/make-a-tea.sh
fi
```

post-commit

```
#!/bin/bash
echo "Popełniono komita"
echo "`git log -1 HEAD`"
```

post-checkout

```
#!/bin/bash
pdflatex main.tex
bibtex main
pdflatex main.tex
pdflatex main.tex
acroread main.pdf &
```


zbierane przez caly rok pomysly studentów i wykadowców
Propozycje oferty dydaktycznej
Glosowanie studentów
Oferta dydaktyczna
Zbieranie preferencji wykadowców
Plan zajeć wraz z przydzialem prowadzących
Zapisy studentów na zajcia
Plany zajeć studentów
Ocena zajeć
Historia zajeć
zapisy na zajcia i ich ocena odbywają sie dwa razy do roku
CYKL ŻYCIA OFERTY DYDAKTYCZNEJ

Ocena zajęć

Obraz pochodzi z systemu zapisów

Serdecznie proszę o szczegółowe opinie!