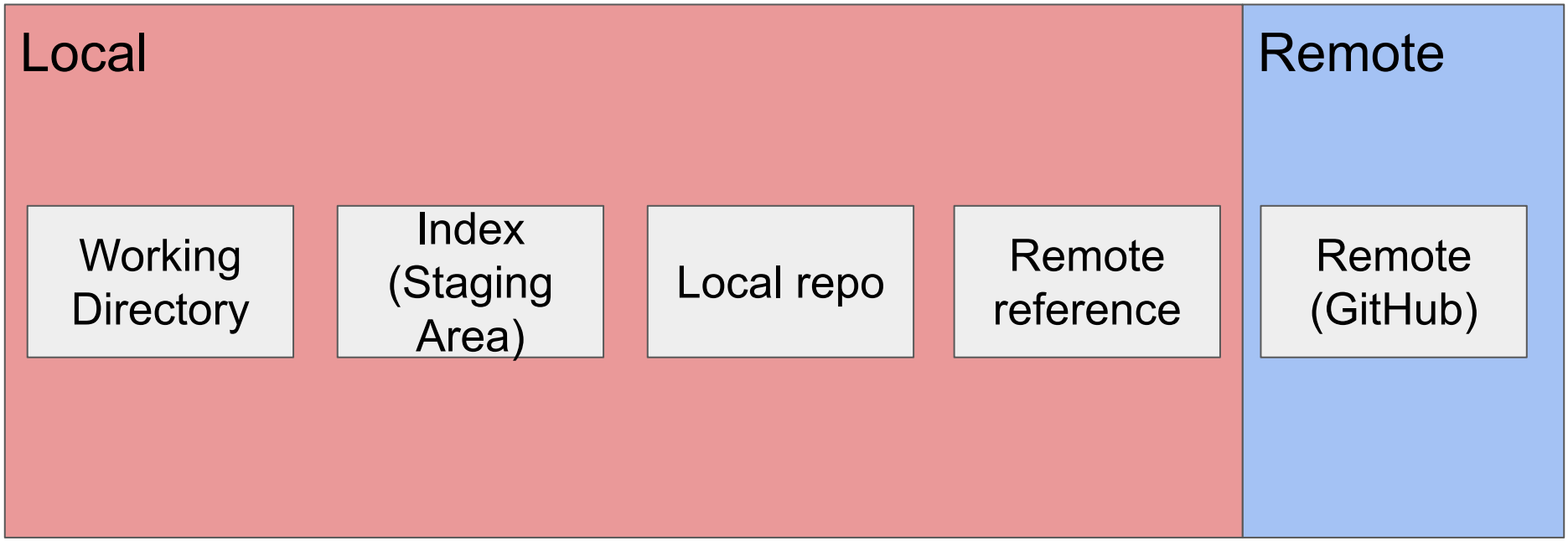
git - local version control

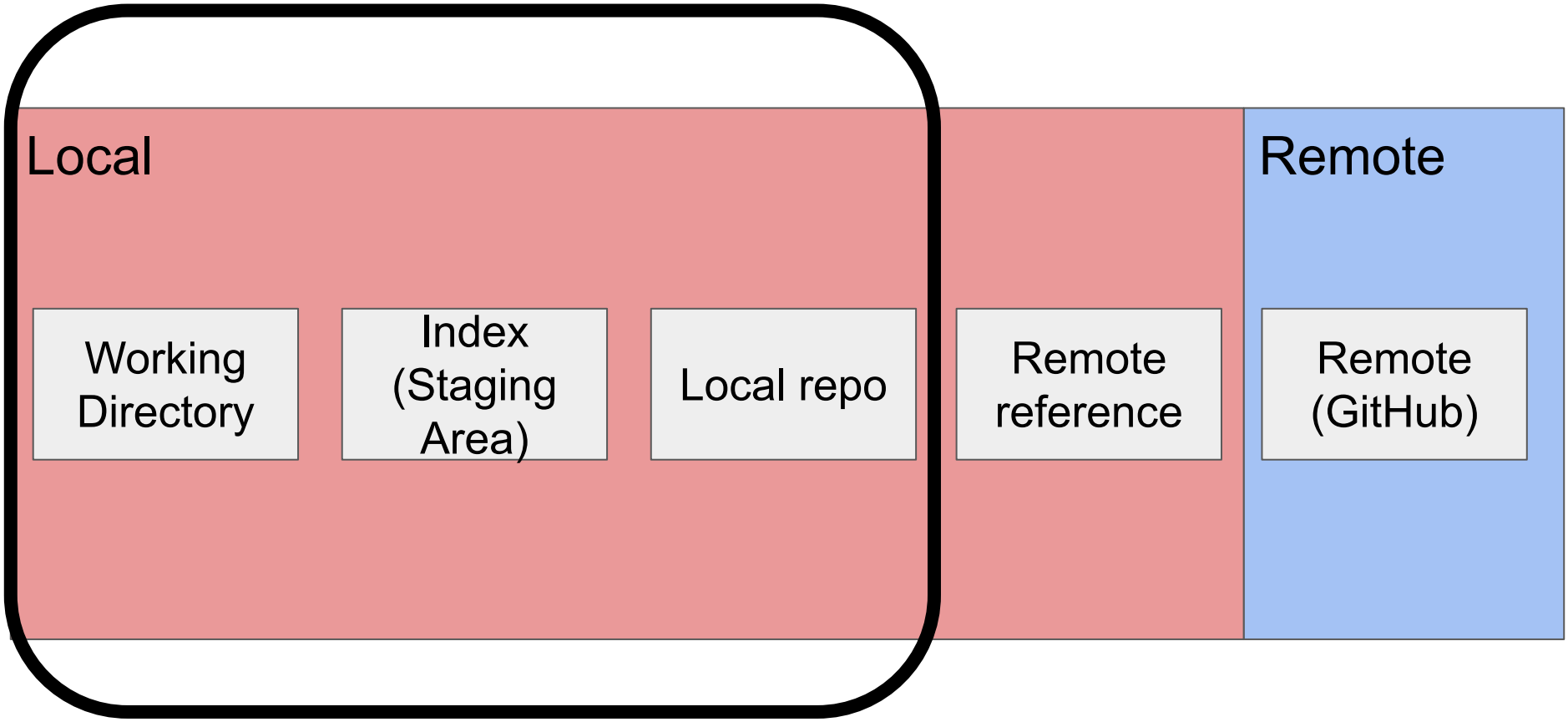
Adapted from materials by Dr. Carrier

Working locally

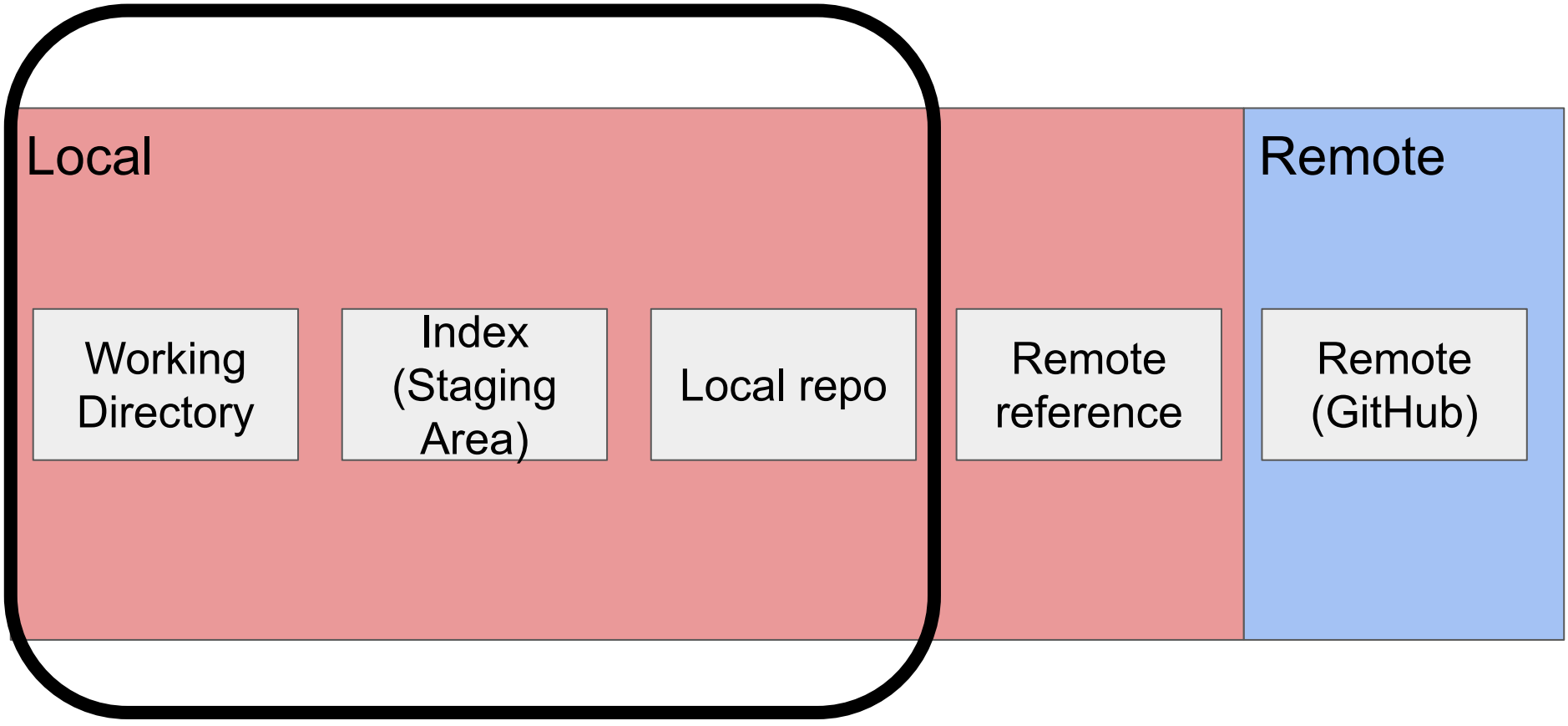
Working locally



Working locally

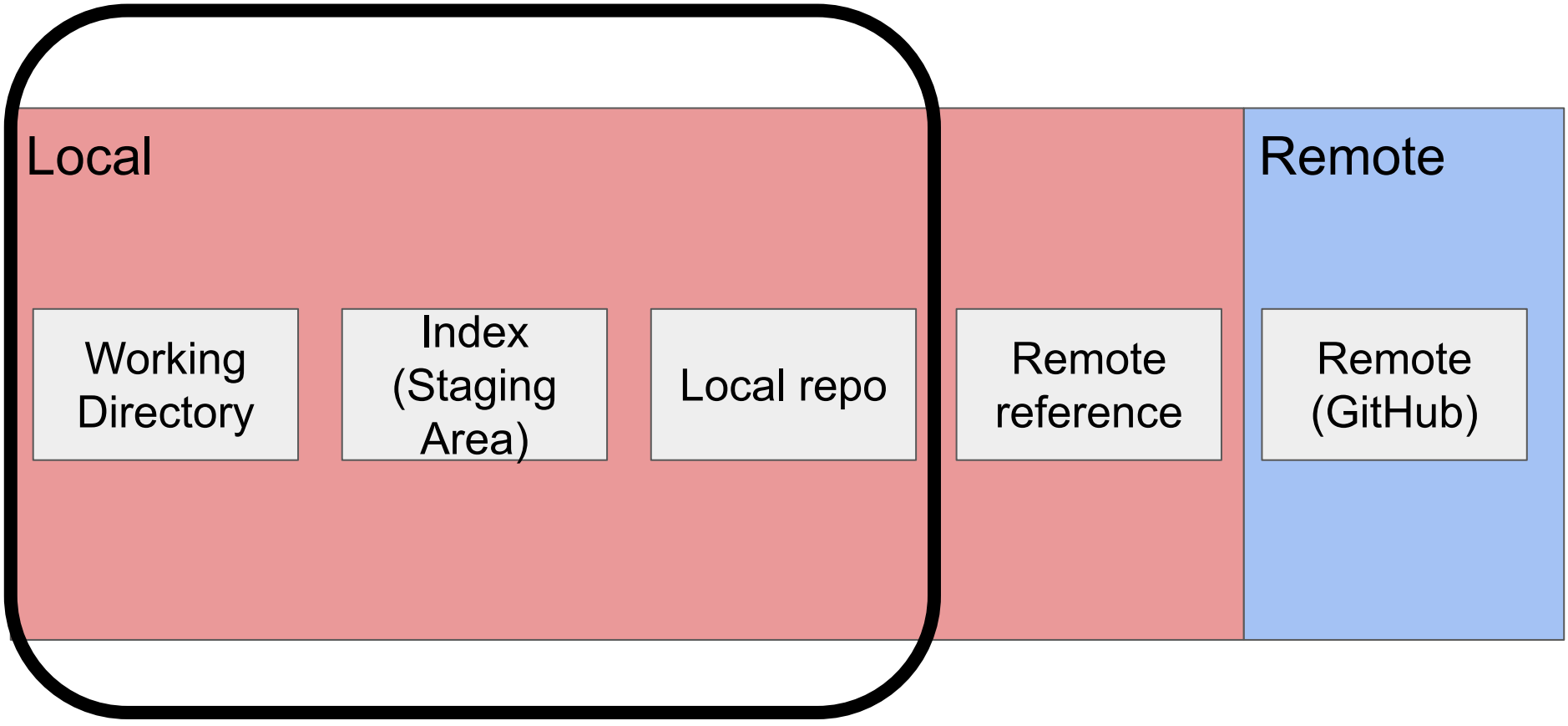


Working locally



What does this give us?

Working locally



What does this give us?
What are we lacking?

Building an analogy

Building an analogy

Imagine you work for IKEA
(Swedish-founded furniture store)

Building an analogy

Imagine you work for IKEA

(Swedish-founded furniture store)((not an ad))

Building an analogy

Imagine you work for IKEA

(Swedish-founded furniture store)((not an ad))

Building an analogy

Imagine you work for IKEA

(Swedish-founded furniture store)((not an ad))

You do not code for IKEA.

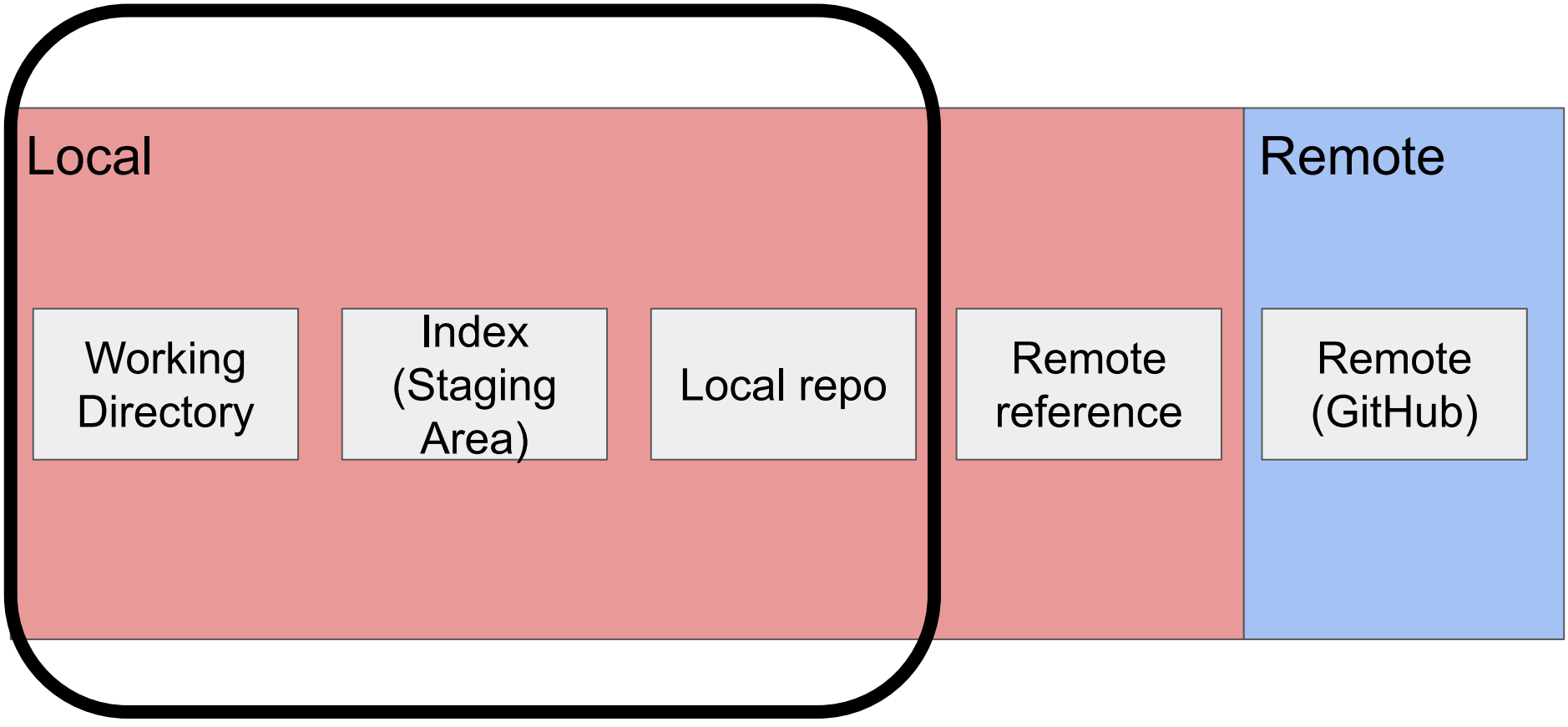
Building an analogy

Imagine you work for IKEA

(Swedish-founded furniture store)((not an ad))

You do not code for IKEA. You make chairs!

Working locally



Taking over the chair industry - Starting up

Taking over the chair industry - Starting up

For now, we will start from scratch

Taking over the chair industry - Starting up

For now, we will start from scratch

Create a new directory

Taking over the chair industry - Starting up

For now, we will start from scratch

Create a new directory

```
git init
```

Taking over the chair industry - Starting up

For now, we will start from scratch

Create a new directory

```
git init
```

This initializes a new git repo

Taking over the chair industry - Starting up

For now, we will start from scratch

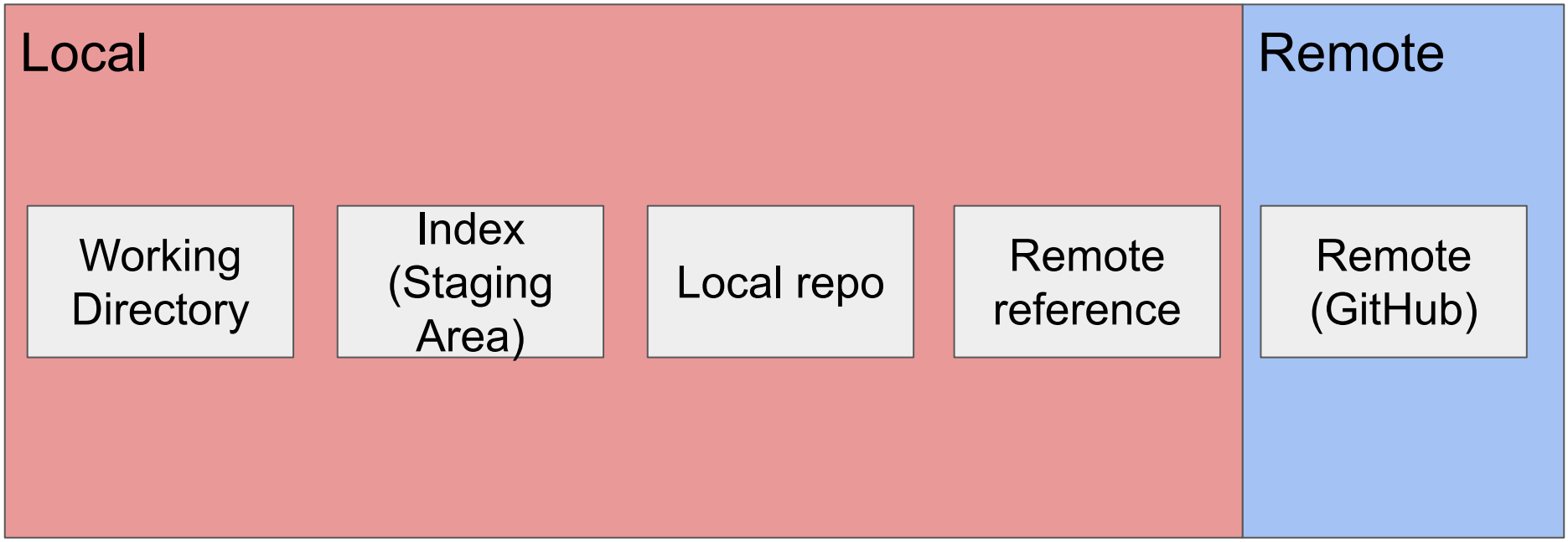
Create a new directory

```
git init
```

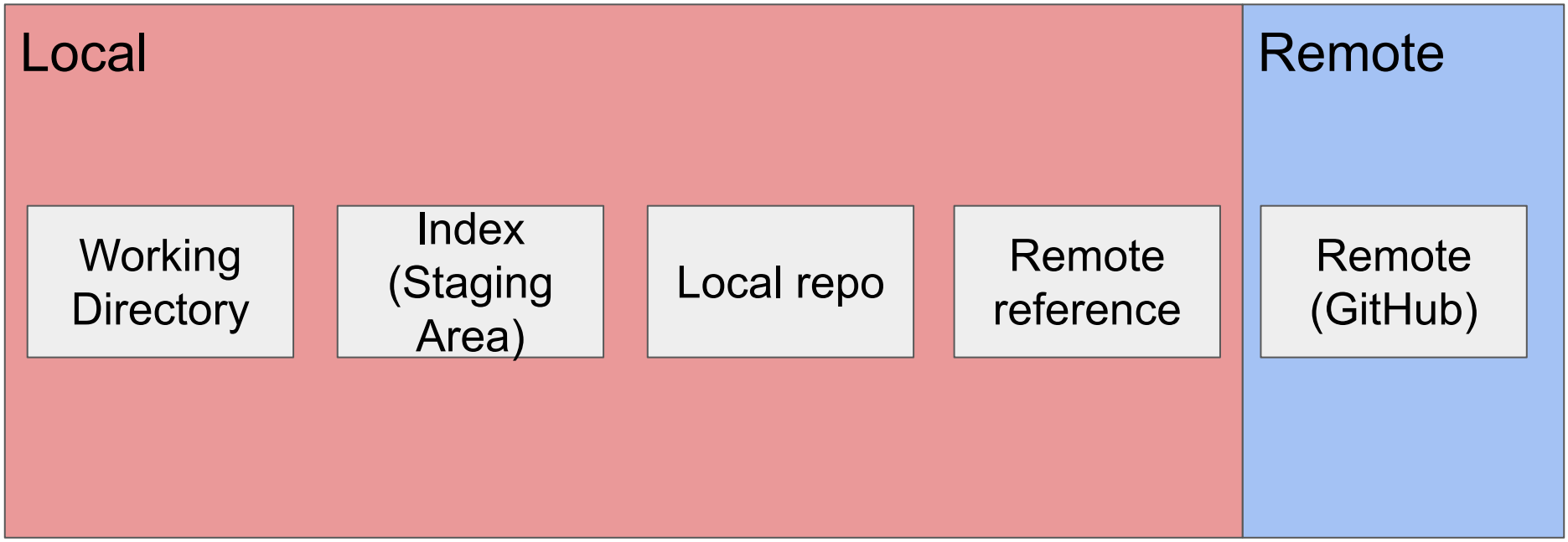
This initializes a new git repo

- The guts are stored in `.git`

The map

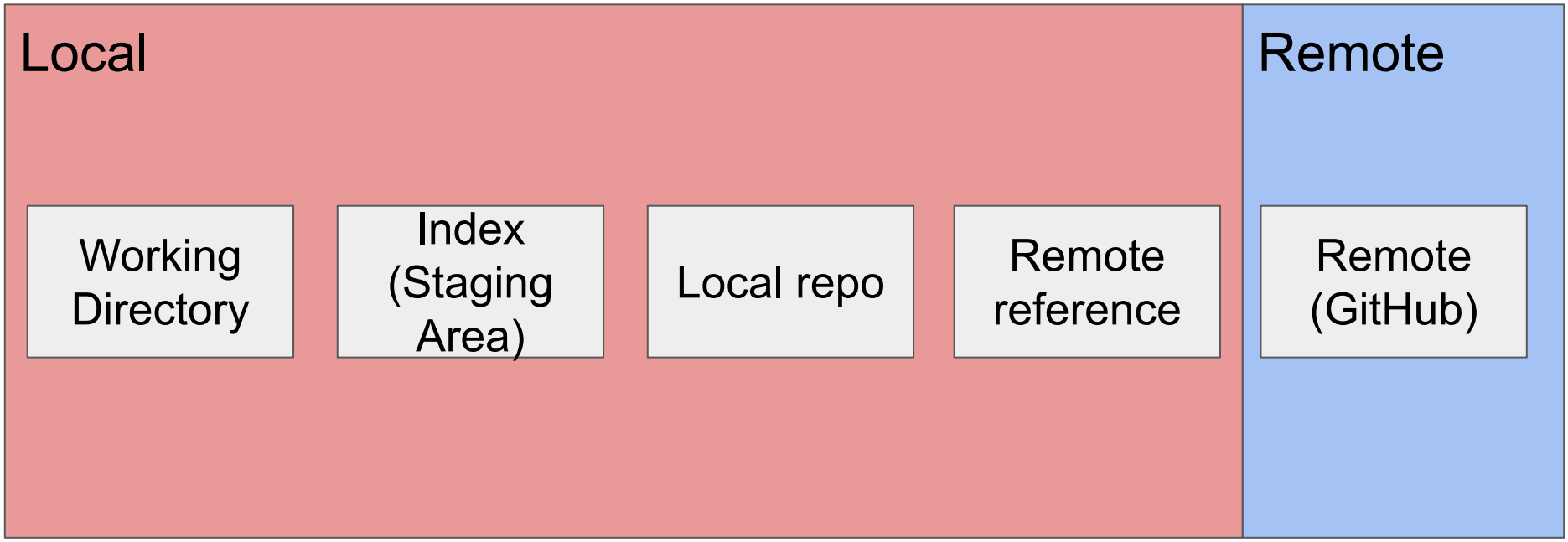

Local
Remote
Working Directory
Index (Staging Area)
Local repo
Remote reference
Remote (GitHub)

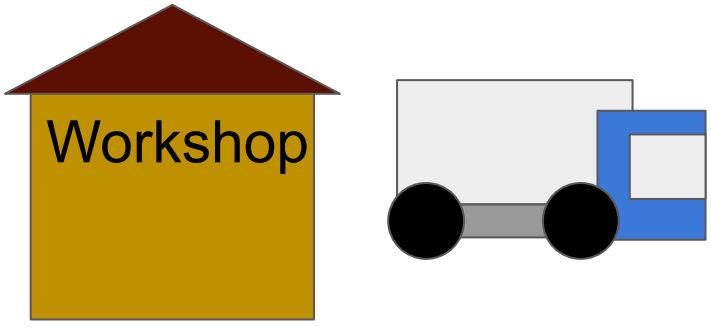
The map


Local
Remote
Working Directory
Index (Staging Area)
Local repo
Remote reference
Remote (GitHub)

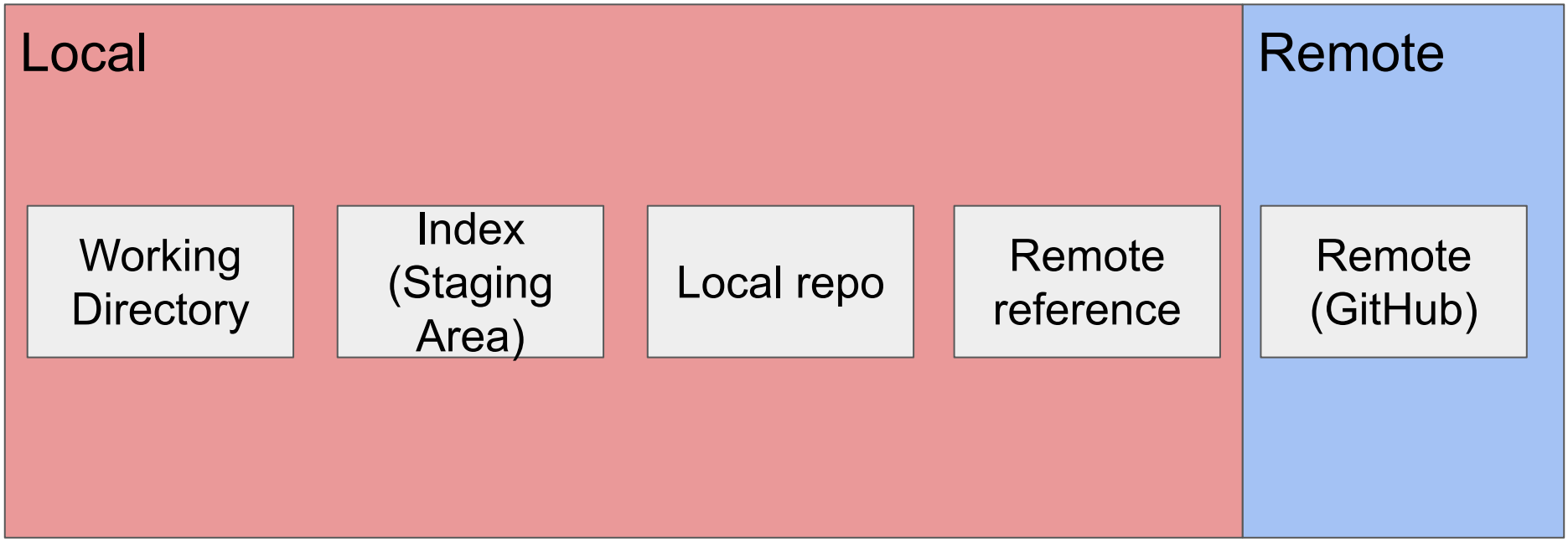

Workshop

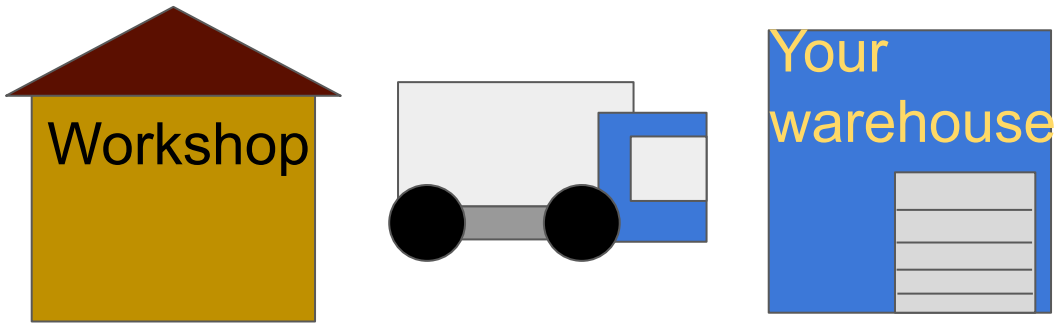
The map


Local
Remote
Working Directory
Index (Staging Area)
Local repo
Remote reference
Remote (GitHub)

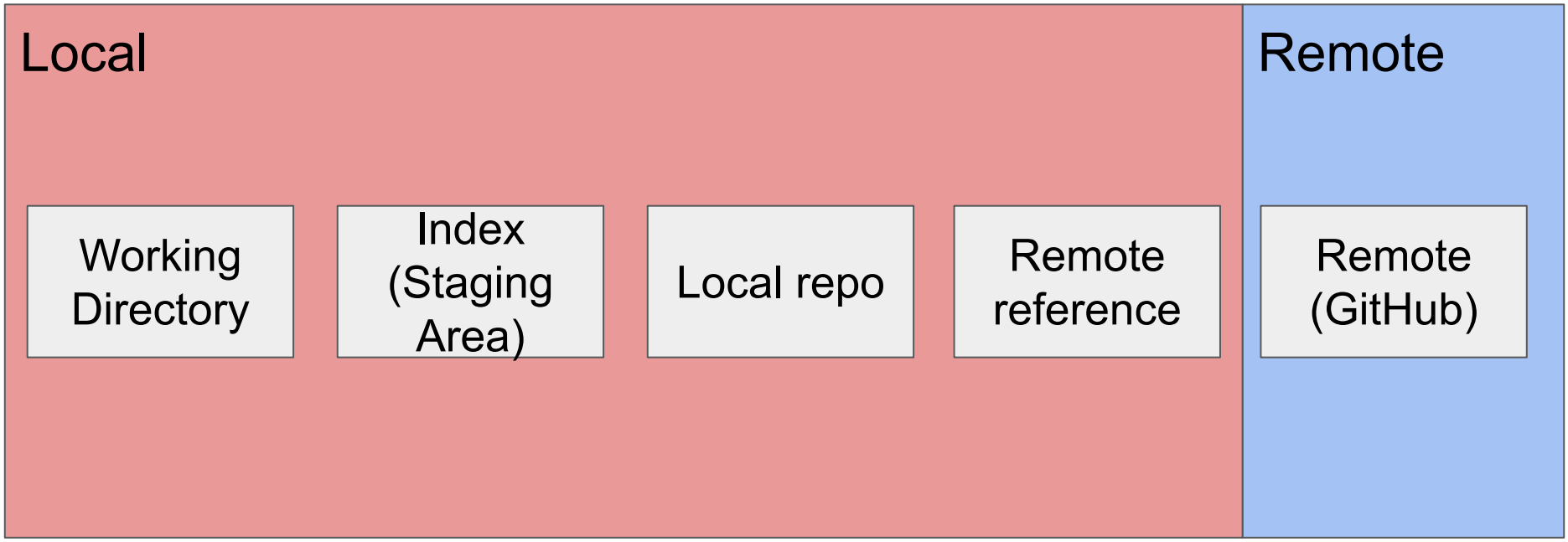

Workshop

The map


Local
Remote
Working Directory
Index (Staging Area)
Local repo
Remote reference
Remote (GitHub)


Workshop
Your warehouse

The map


Local
Remote
Working Directory
Index (Staging Area)
Local repo
Remote reference
Remote (GitHub)


Workshop
Your warehouse
IKEA HQ warehouse

Working directory - Our workshop

Dominik Scythe - Unsplash

Working directory - Our workshop

We can do whatever we want here!

Changes don’t affect local or remote repos

Dominik Scythe - Unsplash

Working directory - Our workshop

We can do whatever we want here!

Changes don’t affect local or remote repos

Unless we tell git to pass along those changes!

Dominik Scythe - Unsplash

How do we track our changes

How do we track our changes

We’ve created a masterpiece!

How do we track our changes

We’ve created a masterpiece!

stefan.txt

How do we track our changes

We’ve created a masterpiece!

stefan.txt

First, we need to stage changes

How do we track our changes

We’ve created a masterpiece!

stefan.txt

First, we need to stage changes

```
git add stefan.txt
```

How do we track our changes

We’ve created a masterpiece!

stefan.txt

First, we need to stage changes

```
git add stefan.txt
```

How do we track our changes

We’ve created a masterpiece!

stefan.txt

First, we need to stage changes

```
git add stefan.txt
```

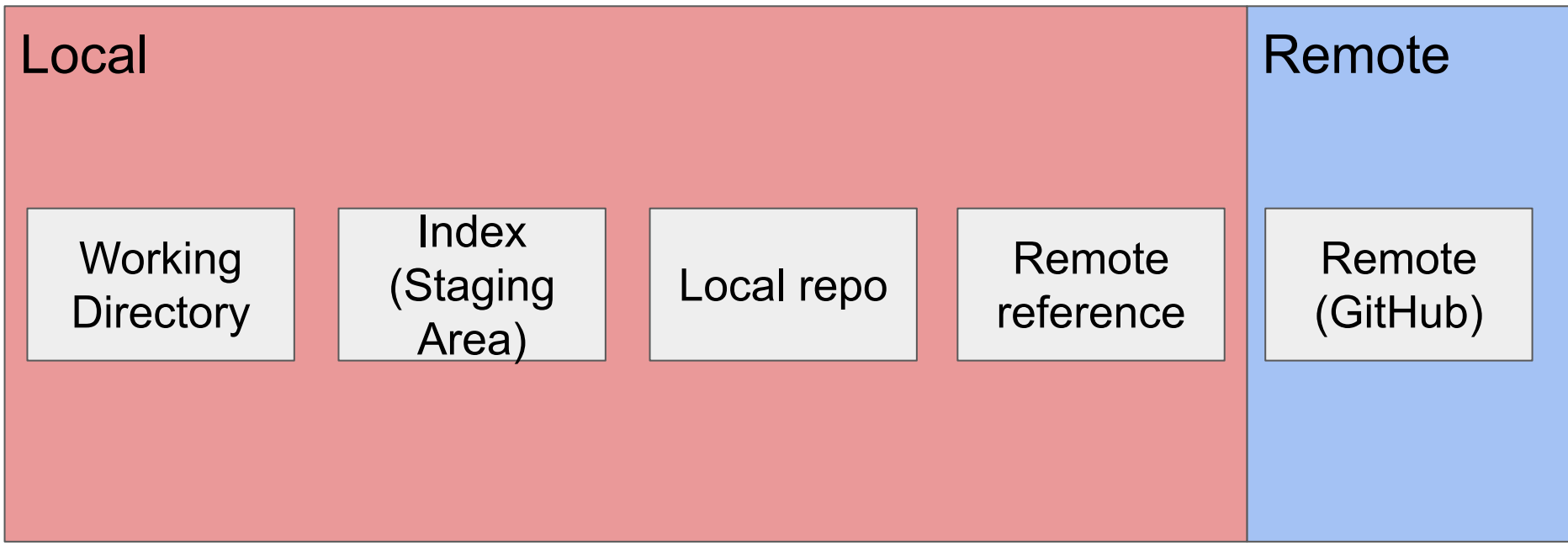
Are the changes saved?

Are the changes saved?

Not yet! They are only staged

Are the changes saved?

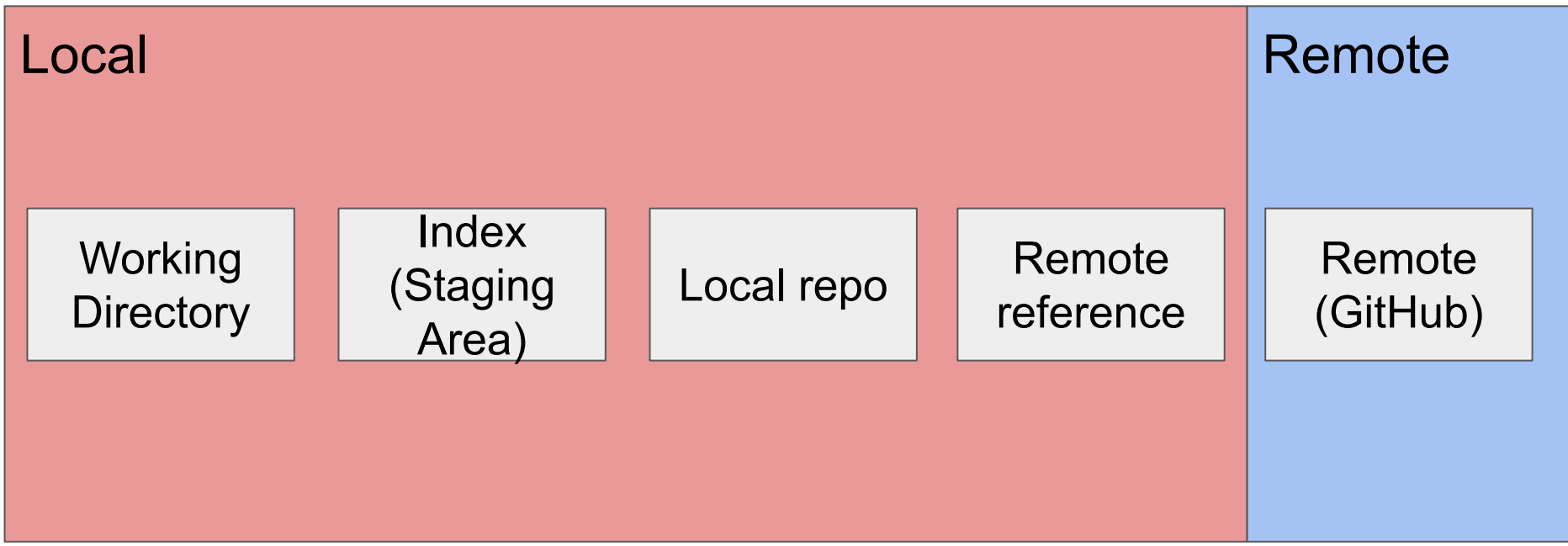
Not yet! They are only staged



Are the changes saved?

Not yet! They are only staged

```
git commit -m “Created stefan”
```



Why do we have the staging area?

Imagine we modified stefan.txt

Why do we have the staging area?

Imagine we modified stefan.txt

Why do we have the staging area?

Imagine we modified stefan.txt

We don’t always want to push
all changes

Why do we have the staging area?

Imagine we modified stefan.txt

We don't always want to push *all* changes

`stefan.txt`

`paint.txt`

Detecting changes

Your new best friend:

Detecting changes

Your new best friend:

```
git status
```

Detecting changes

Your new best friend:

```
git status
```

Git status tells you which files:

- Have been added/changed since last commit
- Are not tracked

Detecting changes

Your new best friend:

```
git status
```

Git status tells you which files:

- Have been added/changed since last commit
- Are not tracked

We can see *how* a file has changed since last commit:

```
git diff
```

Detecting changes

Your new best friend:

```
git status
```

Git status tells you which files:

- Have been added/changed since last commit
- Are not tracked

We can see *how* a file has changed since last commit:

```
git diff
```

Or for a certain file:

```
git diff file
```

Other operations

We can do more than just add files:

Other operations

We can do more than just add files:

```
git mv
```

```
git rm
```

Other operations

We can do more than just add files:

```
git mv
```

```
git rm
```

These do the same things as mv and rm

But now the changes are staged in git!

Viewing history

```
git log
```

This will show the history of commits in our repo

Viewing history

`git log`

This will show the history of commits in our repo

```
ferguaus@eos07:~/git_test$ git log
commit b8e80e582246bfcf985f29707df77b2da35484f8 (HEAD -> master)
Author: Austin Ferguson <ferguaus@eos07.lab.ads.gvsu.edu>
Date:   Thu Sep 19 23:02:22 2024 -0400

    Created stefan

commit 4d9332cefc277f47772af323d3ac123cf6299057
Author: Austin Ferguson <ferguaus@eos07.lab.ads.gvsu.edu>
Date:   Thu Sep 19 21:57:57 2024 -0400

    Initialize repo
```

Viewing history

`git log`

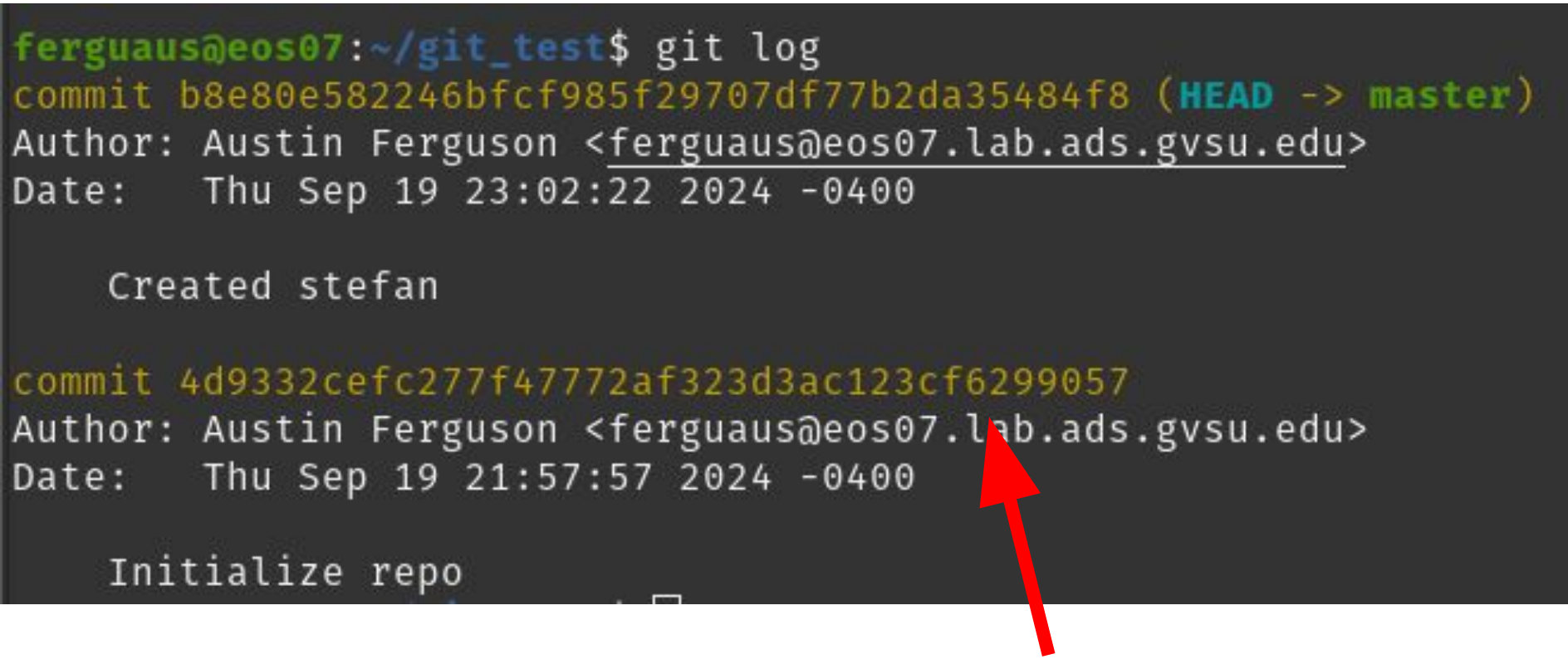
This will show the history of commits in our repo

```
ferguaus@eos07:~/git_test$ git log
commit b8e80e582246bfcf985f29707df77b2da35484f8 (HEAD -> master)
Author: Austin Ferguson <ferguaus@eos07.lab.ads.gvsu.edu>
Date:   Thu Sep 19 23:02:22 2024 -0400

    Created stefan

commit 4d9332cefc277f47772af323d3ac123cf6299057
Author: Austin Ferguson <ferguaus@eos07.lab.ads.gvsu.edu>
Date:   Thu Sep 19 21:57:57 2024 -0400

    Initialize repo
```



SHA-1 hash
Used to identify commits

Correcting mistakes (1)

Help! I’ve accidentally staged a file

Correcting mistakes (1)

Help! I've accidentally staged a file

```
git restore --staged file
```

Or the old way:

```
git reset file
```

Correcting mistakes (2)

Help! I have made a huge mistake and want to get rid of all my changes to a file!

Correcting mistakes (2)

Help! I have made a huge mistake and want to get rid of all my changes to a file!

Correcting mistakes (2)

Help! I have made a huge mistake and want to get rid of all my changes to a file!

First, unstage file if it is staged (previous slide)

Correcting mistakes (2)

Help! I have made a huge mistake and want to get rid of all my changes to a file!

First, unstage file if it is staged (previous slide)

Then:

```
git restore file
```

Or the old way:

```
git checkout file
```

Correcting mistakes (3)

Help! My entire working directory is a lost cause!!!

Correcting mistakes (3)

Help! My entire working directory is a lost cause!!!

Don’t delete untracked files (from top level of repo)

```
git restore --staged --worktree .
```

Correcting mistakes (3)

Help! My entire working directory is a lost cause!!!

Don’t delete untracked files (from top level of repo)

```
git restore --staged --worktree .
```

Or reset changes and possibly **delete untracked files (be careful!)**

```
git reset --hard
```

Cheatsheet

Main file manipulation git commands:

`git init` - Initialize git repo in folder

`git add file` - Add files/changes to staging area (truck)

`git mv source dest` - Move/rename file and stage

`git rm file` - Delete file and stage

`git commit -m “message”` - Commit staged changes to local repo

Status commands

`git status` - View which files changed since last commit

`git diff file` - View how file changed since last commit

`git diff` - View how all files changed

`git log` - View list of commits

Just in case…

STEFAN

New lower price

EKEDALEN

VOXLÖV

PINNTORP