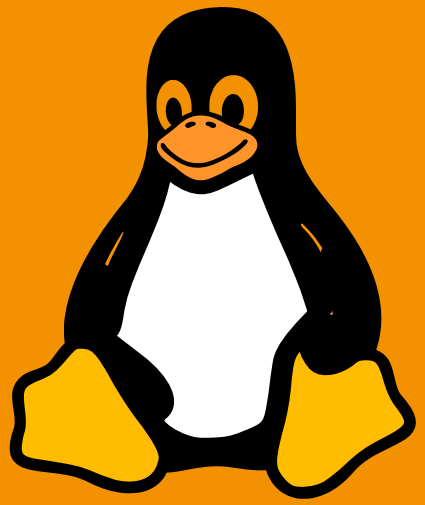
Start: 8:01

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Prof. Dr. Florian Künzner
Technical University of Applied Sciences Rosenheim, Computer Science

Bitte: Vorname + Nachname
⇒ selbständig umbenennen

OS 9 – Communication 2



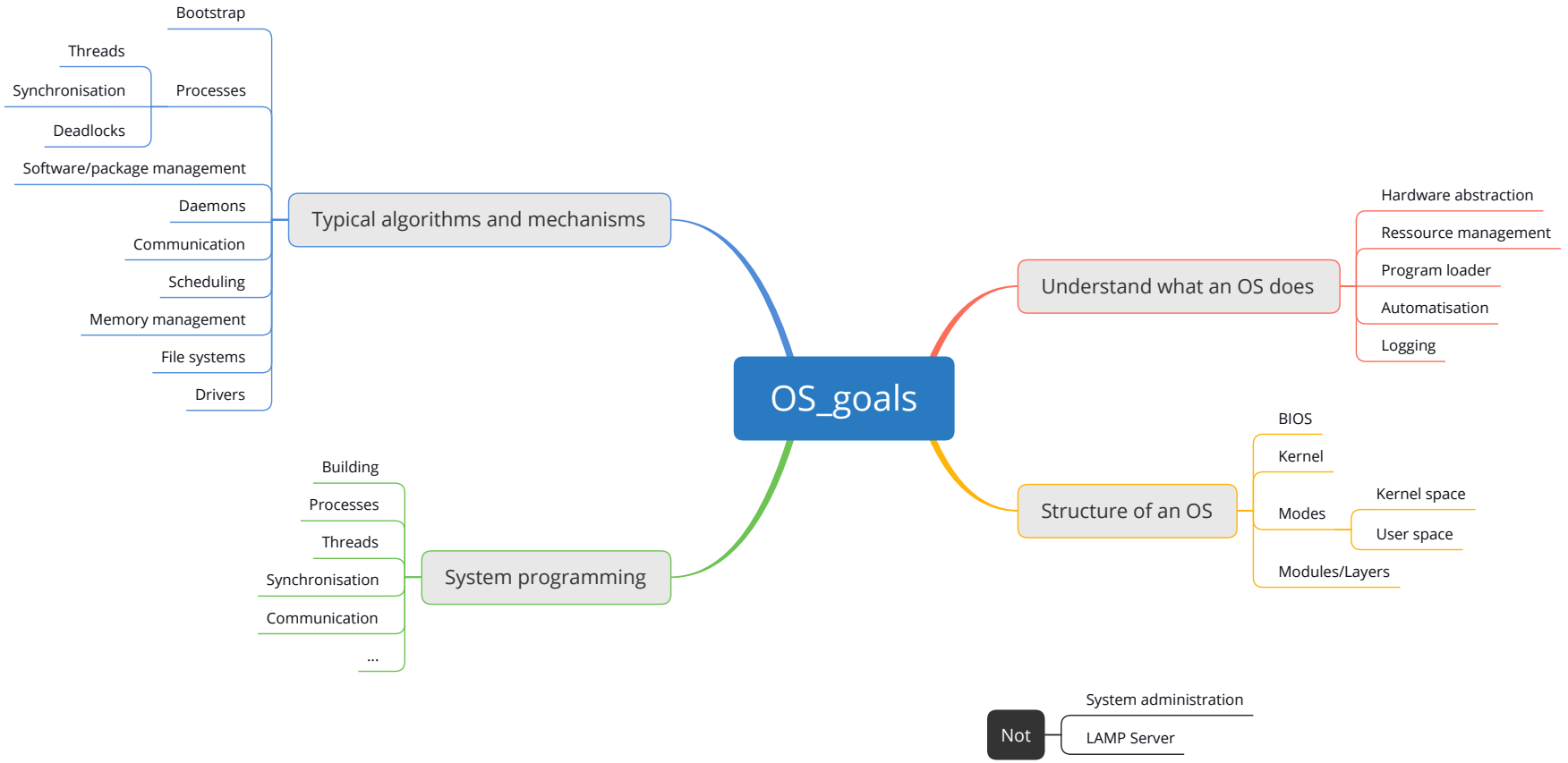
source: iconspng.com

The lecture is based on the work and the documents of Prof. Dr. Ludwig Frank

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Goal



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Goal

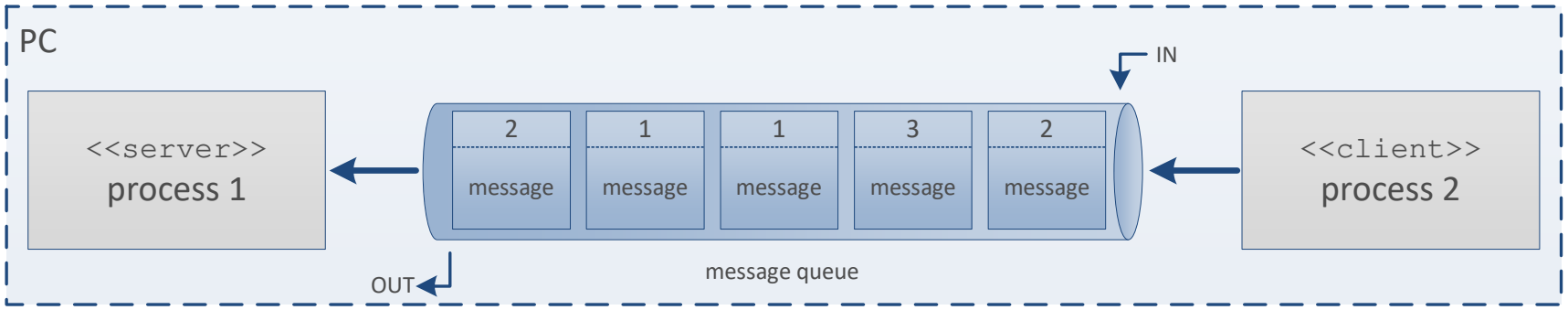
OS::Communication

- Message queue
- Shared memory
- Process communication summary

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message queue concept

- Queue to **store messages**
- Inter-process communication (IPC) between processes on one PC.
- Messages have **priority/type**.
- Internal stored as a linked list.
- **Send** into queue **does not require** an **active receiver**
- **Read** from queue **does not require** an **active sender**
- Max queue size (default: 16 KiB on Linux).
- Max message size (default: 8 KiB on Linux).

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

0 is not allowed!
1 highest prio
>1 lower prio

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (PRIO_FETCH_FLAG)

- Read message after message (FIFO principle) (priority == 0)
- Read only message with specific priority/type (priority == N)
- Read the messages with the highest priority/type first, up to a certain number (priority == -N)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (PRIO_FETCH_FLAG)

- Read message after message (FIFO principle) (priority == 0)
- Read only message with specific priority/type (priority == N)
- Read the messages with the highest priority/type first, up to a certain number (priority == -N)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

Message structure

```
1 struct message {
2     long priority;     //priority or type
3     char message[64]; //buffer for message bytes
4 };
```

Message priority/type

- The lower the number, the higher the priority
- The priority can be interpreted as a type (each type has its own number)

Message queue usage scenarios (`PRIO_FETCH_FLAG`)

- Read message after message (FIFO principle) (`priority == 0`)
- Read only message with specific priority/type (`priority == N`)
- Read the messages with the highest priority/type first, up to a certain number (`priority == -N`)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue: Pseudo C code

```c
struct message { //structure for messages
  long priority;
  char message[64];
};

void receiver() {
  //create message queue
  msgget(...);


  //receive message
  //blocks if message queue is empty
  msgrcv(...);

  //... work with message

  //remove message queue
  msgctl(...);
}

void sender() {
  //open existing message queue
  msgget(...);

  //prepare message

  //send message
  //blocks if message queue is full
  msgsnd(...);

  //close not needed


}
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue: Pseudo C code

```c
struct message { //structure for messages
  long priority;
  char message[64];
};

void receiver() {
  //create message queue
  msgget(...);



  //receive message
  //blocks if message queue is empty
  msgrcv(...);

  //... work with message

  //remove message queue
  msgctl(...);
}
```

```c
void sender() {
  //open existing message queue
  msgget(...);

  //prepare message

  //send message
  //blocks if message queue is full
  msgsnd(...);

  //close not needed


}
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue: Pseudo C code

```c
struct message { //structure for messages
  long priority;
  char message[64];
};

void receiver() {
  //create message queue
  msgget(...);



  //receive message
  //blocks if message queue is empty
  msgrcv(...);

  //... work with message

  //remove message queue
  msgctl(...);
}
```

```c
void sender() {
  //open existing message queue
  msgget(...);

  //prepare message

  //send message
  //blocks if message queue is full
  msgsnd(...);

  //close not needed


}
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue: Linux commands

Command	Description
`ipcs`	Show information on IPC facilities
`ipcs -q`	Shows active message queues in the system
`ipcmk`	Make various IPC resources
`ipcmk -Q`	Create a message queue
`ipcrm`	Remove certain IPC resources
`ipcrm -q 1`	Remove message queue with id 1
`ipcrm -Q 2`	Remove message queue with key 2

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Message queue

C example

*Please find the source file(s) in the repository.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? ⇒ 

Question? ⇒  and use **chat**

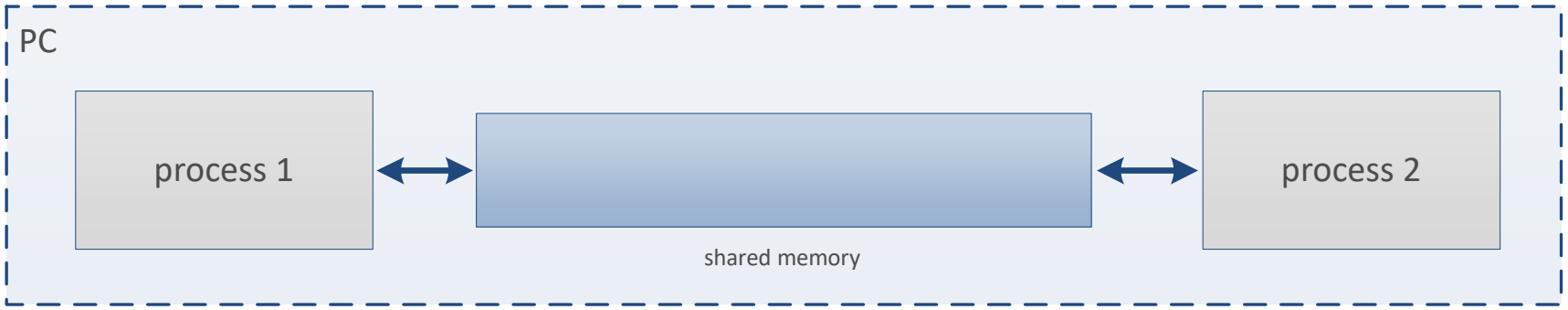
or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

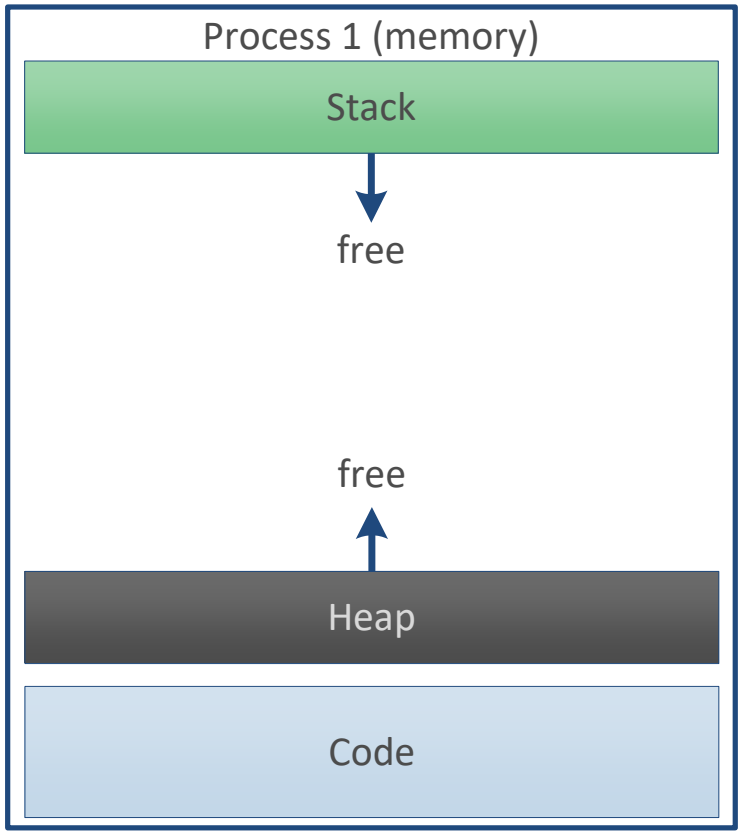
Shared memory



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

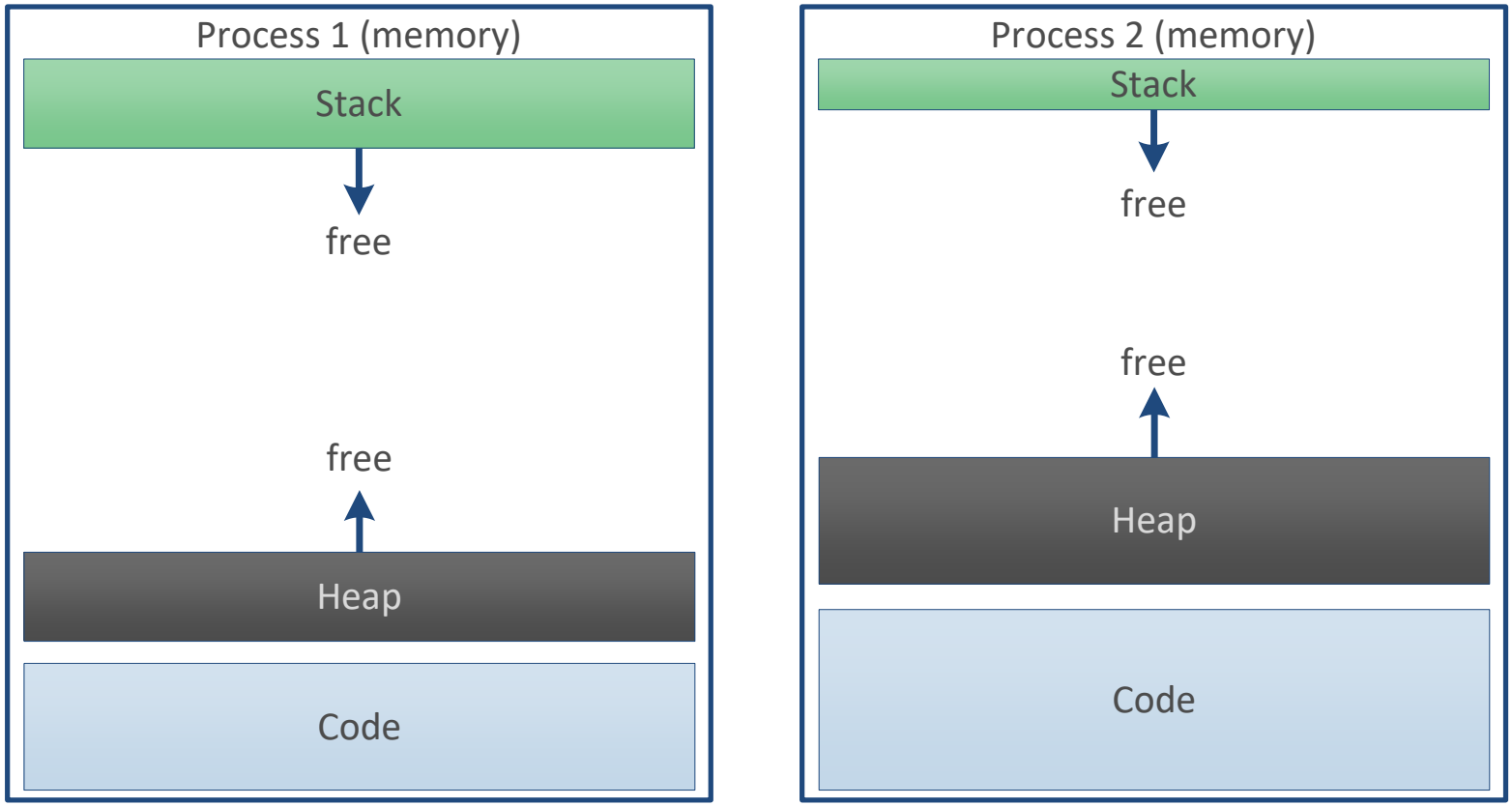
Shared memory



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

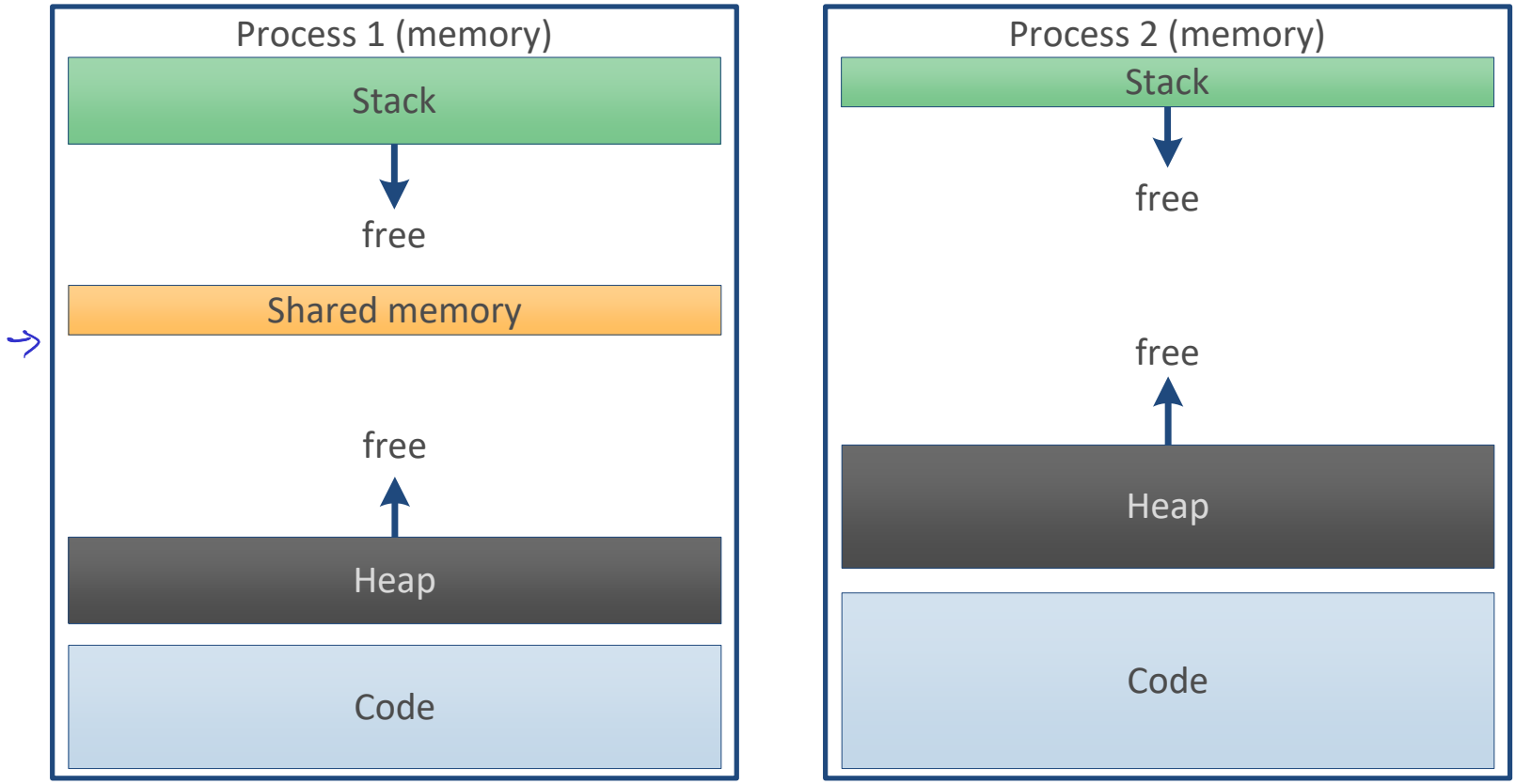
Shared memory



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

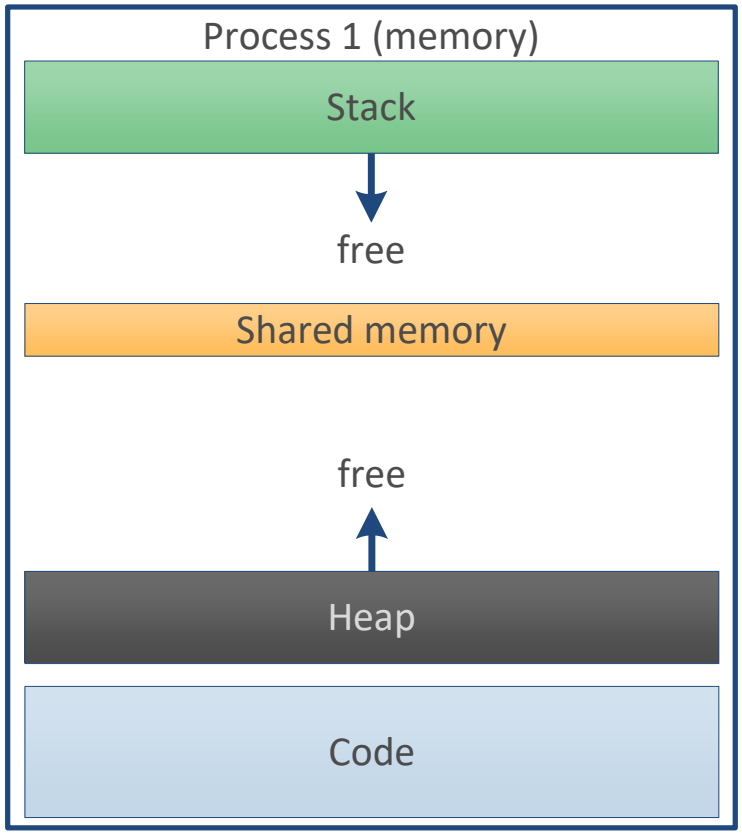
Shared memory

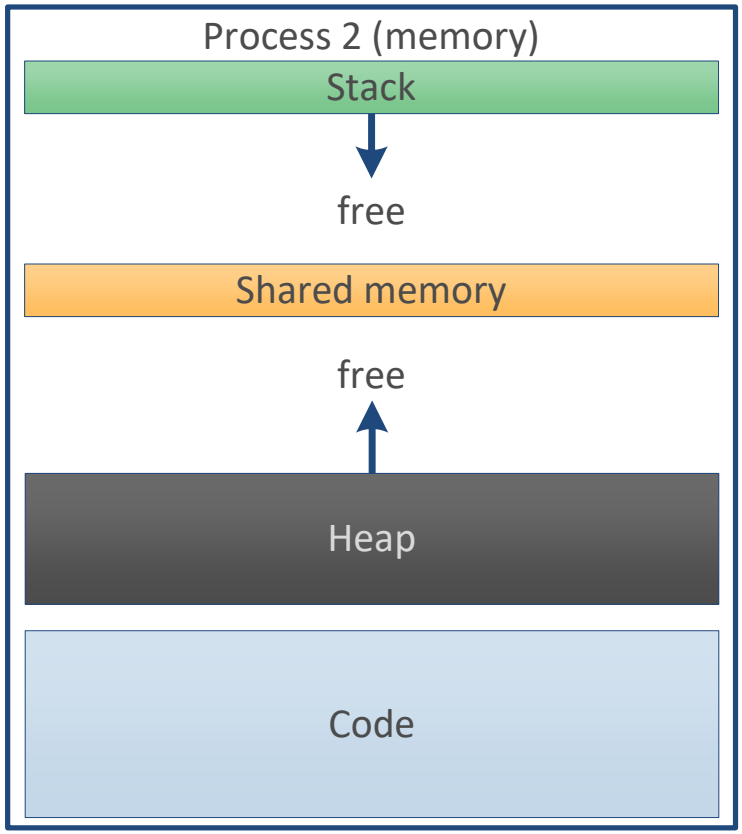


CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

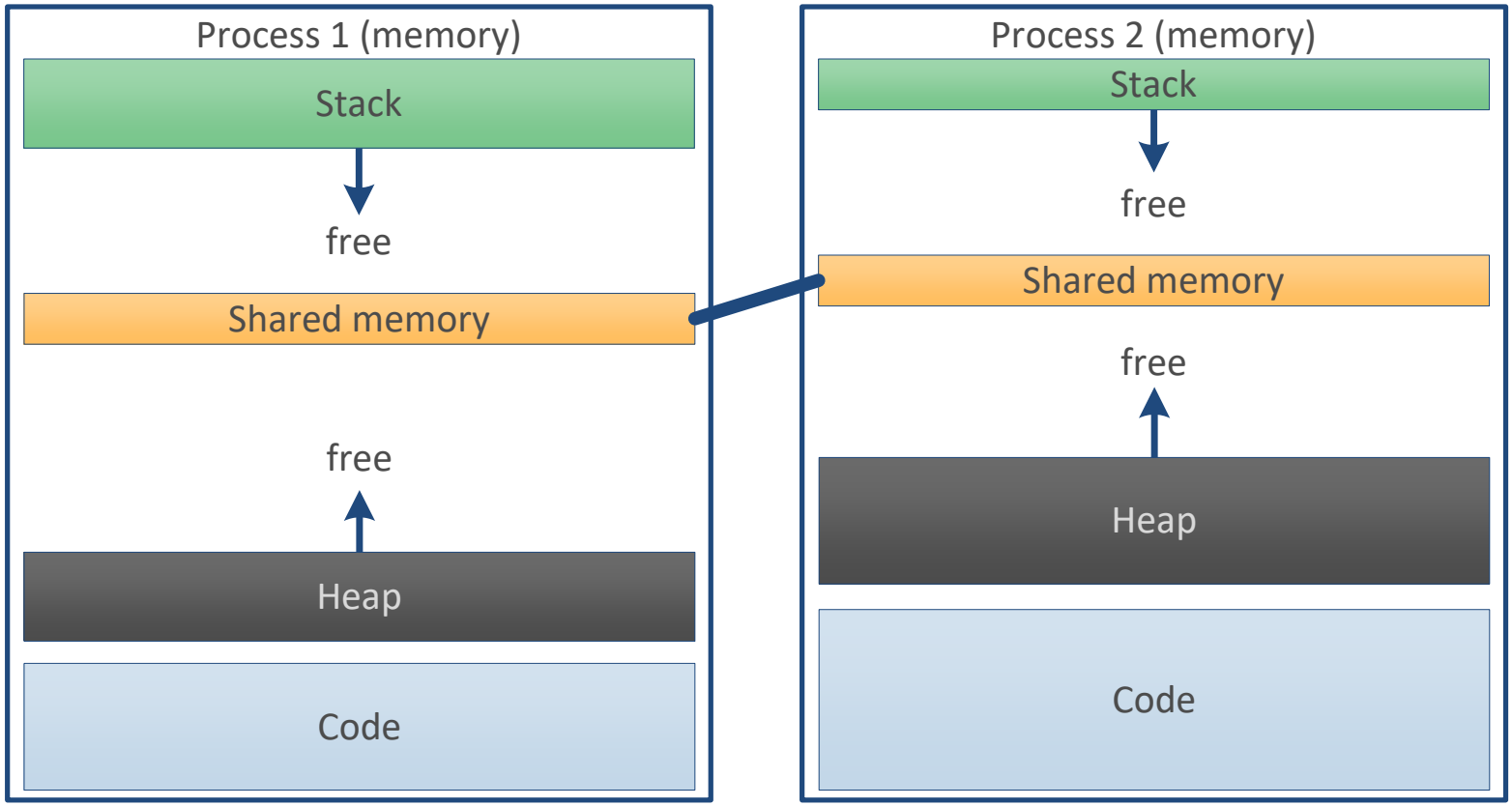




CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory



CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

Shared memory concept

- Shared memory area between processes
- Inter-process communication (IPC) between processes on one PC.
- It is a **plain memory area** with a certain size
- Access **needs to be synchronised** (e.g. semaphore)
- Access is **very fast** (comparable with own memory access)

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory: Pseudo C code

```
1  seminit(READY_TO_WRITE, 1); //declare and initialise semaphore
2  seminit(READY_TO_READ,  0); //declare and initialise semaphore
3  void receiver() {
4    //create shared memory
5    shmget(...);
6    //attach the shared memory
7    shared_mem_address = shmat(...);
8
9
10
11
12   //copy data from shared memory
13   P(READY_TO_READ);
14   copy(data, shared_mem_address); //data = sm
15   V(READY_TO_WRITE);
16
17   //... work with data
18   work_with(data);
19
20   //detach shared memory
21   shmdt(...);
22   //remove shared memory
23   shmctl(...);
24 }
25 void sender() {
26   //get existing shared memory
27   shmget(...);
28   //attach the shared memory
29   shared_mem_address = shmat(...);
30
31   //... prepare data
32   data = prepare_data();
33
34   //copy data into shared memory
35   P(READY_TO_WRITE);
36   copy(shared_mem_address, data); //sm = data
37   V(READY_TO_READ);
38
39
40
41
42   //detach shared memory
43   shmdt(...);
44
45
46 }
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory: Pseudo C code

```
1  seminit(READY_TO_WRITE, 1); //declare and initialise semaphore
2  seminit(READY_TO_READ,  0); //declare and initialise semaphore

3  void receiver() {
4    //create shared memory
5    shmget(...);
6    //attach the shared memory
7    shared_mem_address = shmat(...);
8
9
10
11
12   //copy data from shared memory
13   P(READY_TO_READ);
14   copy(data, shared_mem_address); //data = sm
15   V(READY_TO_WRITE);
16
17   //... work with data
18   work_with(data);
19
20   //detach shared memory
21   shmdt(...);
22   //remove shared memory
23   shmctl(...);
24 }
```

```
25 void sender() {
26   //get existing shared memory
27   shmget(...);
28   //attach the shared memory
29   shared_mem_address = shmat(...);
30
31   //... prepare data
32   data = prepare_data();
33
34   //copy data into shared memory
35   P(READY_TO_WRITE);
36   copy(shared_mem_address, data); //sm = data
37   V(READY_TO_READ);
38
39
40
41
42   //detach shared memory
43   shmdt(...);
44
45
46 }
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory: Pseudo C code

```
1  seminit(READY_TO_WRITE, 1); //declare and initialise semaphore
2  seminit(READY_TO_READ,  0); //declare and initialise semaphore

3  void receiver() {
4    //create shared memory
5    shmget(...);
6    //attach the shared memory
7    shared_mem_address = shmat(...);
8
9
10
11
12   //copy data from shared memory
13   P(READY_TO_READ);
14   copy(data, shared_mem_address); //data = sm
15   V(READY_TO_WRITE);
16
17   //... work with data
18   work_with(data);
19
20   //detach shared memory
21   shmdt(...);
22   //remove shared memory
23   shmctl(...);
24 }

25 void sender() {
26   //get existing shared memory
27   shmget(...);
28   //attach the shared memory
29   shared_mem_address = shmat(...);
30
31   //... prepare data
32   data = prepare_data();
33
34   //copy data into shared memory
35   P(READY_TO_WRITE);
36   copy(shared_mem_address, data); //sm = data
37   V(READY_TO_READ);
38
39
40
41
42   //detach shared memory
43   shmdt(...);
44
45
46 }
```

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory: Linux commands

Command	Description
`ipcs`	Show information on IPC facilities
`ipcs -m`	Shows active shared memory in the system
`ipcmk`	Make various IPC resources
`ipcmk -M 8`	Create a shared memory with 8 bytes
`ipcrm`	Remove certain IPC resources
`ipcrm -m 1`	Remove shared memory with id 1
`ipcrm -M 2`	Remove shared memory with key 2

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Shared memory

C example

*Please find the source file(s) in the repository.

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? ⇒ 

Question? ⇒  and use **chat**

or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal								
unix socket								
network socket								
message queue								
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket								
network socket								
message queue								
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket	X		X		X	X		
network socket								
message queue								
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket	X		X		X	X		
network socket	X			X	X			
message queue								
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket	X		X		X	X		
network socket	X			X	X			
message queue								
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket	X		X		X	X		
network socket	X			X	X			
message queue	X	X	X			X	X	
shared memory								

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Process communication summary

Mechanism	data	store	access contr.	remote	bidirect.	fast	prio.	sync req.
signal						X		
unix socket	X		X		X	X		
network socket	X			X	X			
message queue	X	X	X			X	X	
shared memory	XX	XX	X		X	XX		X

Comparison of different communication mechanism: `https://www.programering.com/a/MTO0AzMwATI.html`

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Questions?

All right? ⇒ 

Question? ⇒  and use **chat**

or

speak *after* I ask you to

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Summary and outlook

Summary

- Message queue
- Shared memory
- Process communication summary

Outlook

- Deadlocks
- Deadlock analysis
- Deadlock prevention

CAMPUS Rosenheim
Computer Science

Technische Hochschule Rosenheim
Technical University of Applied Sciences

Summary and outlook

Summary

- Message queue
- Shared memory
- Process communication summary

Outlook

- Deadlocks
- Deadlock analysis
- Deadlock prevention