제07장

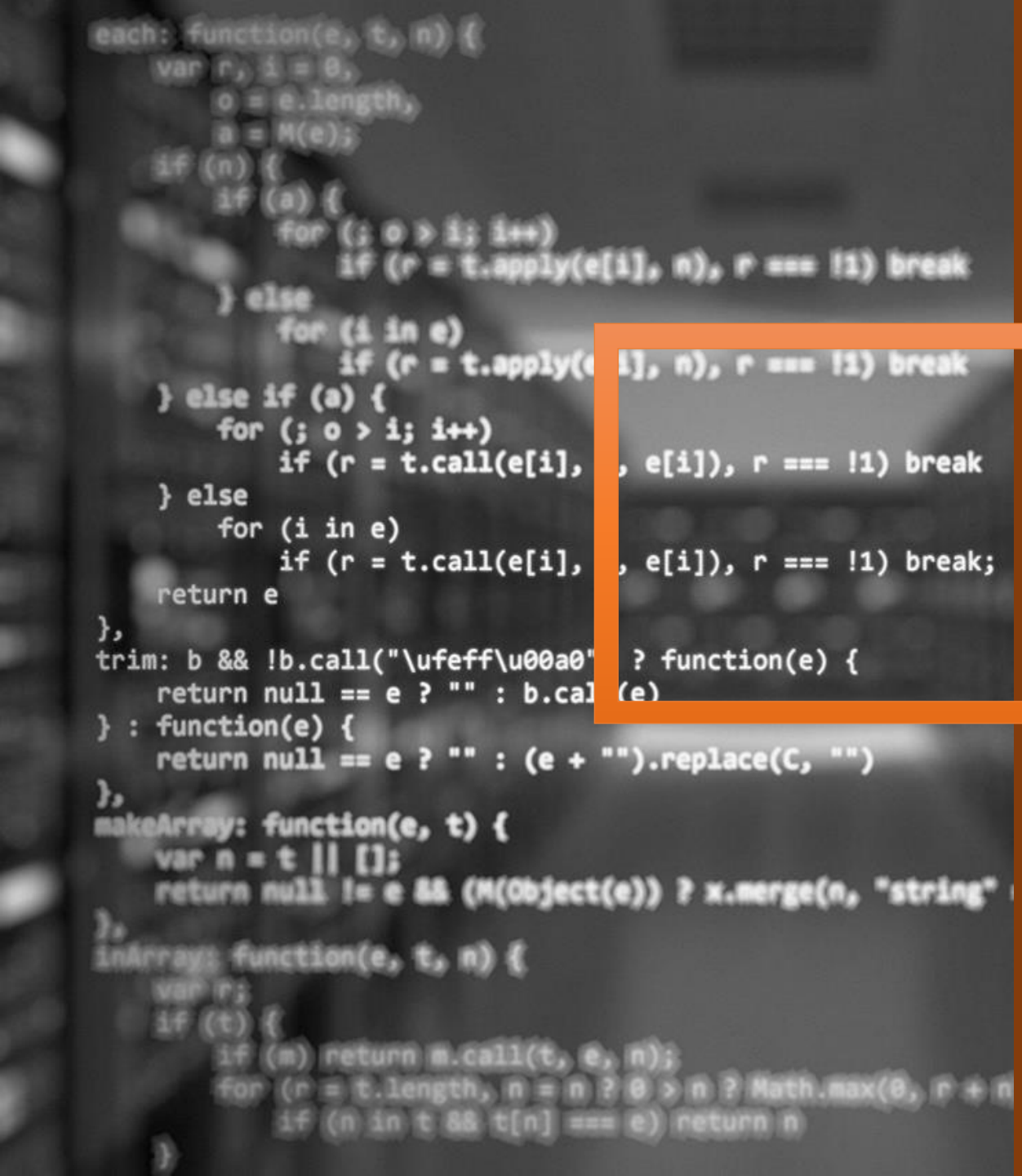
Docker 명령어

DevOps

학습목표

1. 도커 명령어 구조를 이해할 수 있다.
2. 도커 이미지 명령어를 이해하고 실행할 수 있다.
3. 도커 컨테이너 명령어를 이해하고 실행할 수 있다.
4. 도커 볼륨 명령어를 이해하고 실행할 수 있다.

목차



01. 도커 명령어 구조

도커 기본 명령어 구조

고정 명령	상위 명령	하위 명령	옵션	대상	인자
	무엇을	어떻게	명령어 세부 설정	명령어 수행 대상	대상에 전달할 값
docker	container	start stop run ls exec ...	-a -d -i -o -p -q -v ...		
	image	create pull ls search build ...			
	volume	create ls rm ...			
	network	create ls rm ...			

도커 상위 명령

- container
  - 도커 컨테이너를 다루는 명령
  - 도커 컨테이너를 실행하면 애플리케이션이 실행됨

- image
  - 도커 이미지를 다루는 명령
  - 도커 컨테이너를 만들기 위한 구성 요소를 담고 있으며 Dockerfile을 이용해서 생성
  - 도커 허브에 공개된 도커 이미지를 언제든 가져다 사용할 수 있음

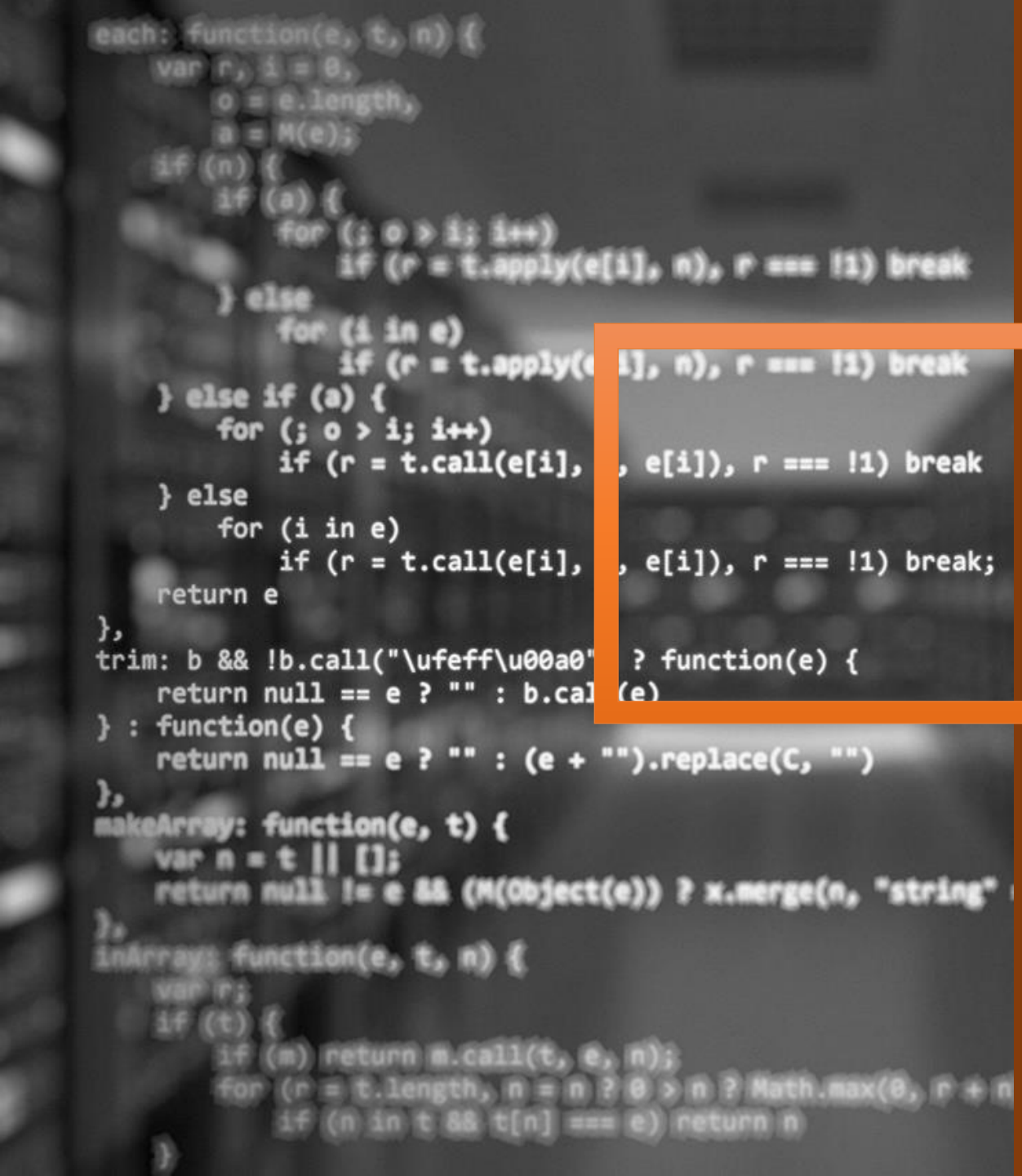
도커 상위 명령

- volume
  - 도커 볼륨을 다루는 명령
  - 컨테이너 내부 데이터를 컨테이너 외부 저장소와 연결하는 기능
  - 디렉터리를 직접 연결하는 바인드 마운트 방식과 도커 볼륨이 관리하는 볼륨 마운트 방식으로 구분

- network
  - 도커 네트워크를 다루는 명령
  - 컨테이너간의 통신을 관리하고 격리하기 위한 기능
  - 컨테이너간 연결이 필요한 경우 동일한 네트워크로 묶어서 사용

도커 하위 명령

- create
  - 생성
- rm
  - 삭제
- ls
  - 목록 보기
- start
  - 시작
- stop
  - 중지
- inspect
  - 상세 조회
- run
  - 실행하기
- search
  - 검색하기



02. 도커 이미지 명령어

docker image

- docker image 하위명령 [옵션]

하위명령	내용
pull	Docker Hub와 같은 도커 이미지 레지스트리에서 이미지를 내려 받는 명령어
ls	저장된 이미지들의 목록을 출력하는 명령어
inspect	특정 이미지의 상세 정보를 조회하는 명령어
build	Dockerfile을 실행하여 도커 이미지를 생성하는 명령어
tag	이미지에 별칭을 부여하는 명령어
rm	이미지를 삭제하는 명령어

docker image

- docker image pull [옵션] 이미지명
  - Docker Hub 같은 레지스트리로부터 이미지를 내려 받는 명령어

옵션	내용
-a, --all-tags	Docker Hub의 모든 버전 이미지 다운로드
--disable-content-trust	이미지 확인 건너뛰기 옵션 / true(기본값), false
-q, --quiet	출력 메세지 차단

docker image

- docker image ls [옵션] / docker images
  - 저장되어 있는 이미지들의 목록을 출력하는 명령어

옵션	내용
-a, --all	모든 이미지 표시
--digests	다이제스트 표시 (다이제스트: 이미지 위변조를 방지하기 위한 해시값)
--no-trunc	각 결과 항목을 자르지 않고 모두 출력
-q, --quiet	도커 이미지 아이디만 표시

docker image

- docker image inspect 이미지명
  - 이미지의 상세 정보를 출력하는 명령어

옵션	내용
--format	출력 형식을 지정해서 특정 값만 출력

docker image

- docker image rm 이미지명 [옵션] / docker rmi
  - 저장된 이미지를 삭제하는 명령어
  - 해당 이미지로 생성한 컨테이너가 실행 중인 경우 삭제할 수 없음

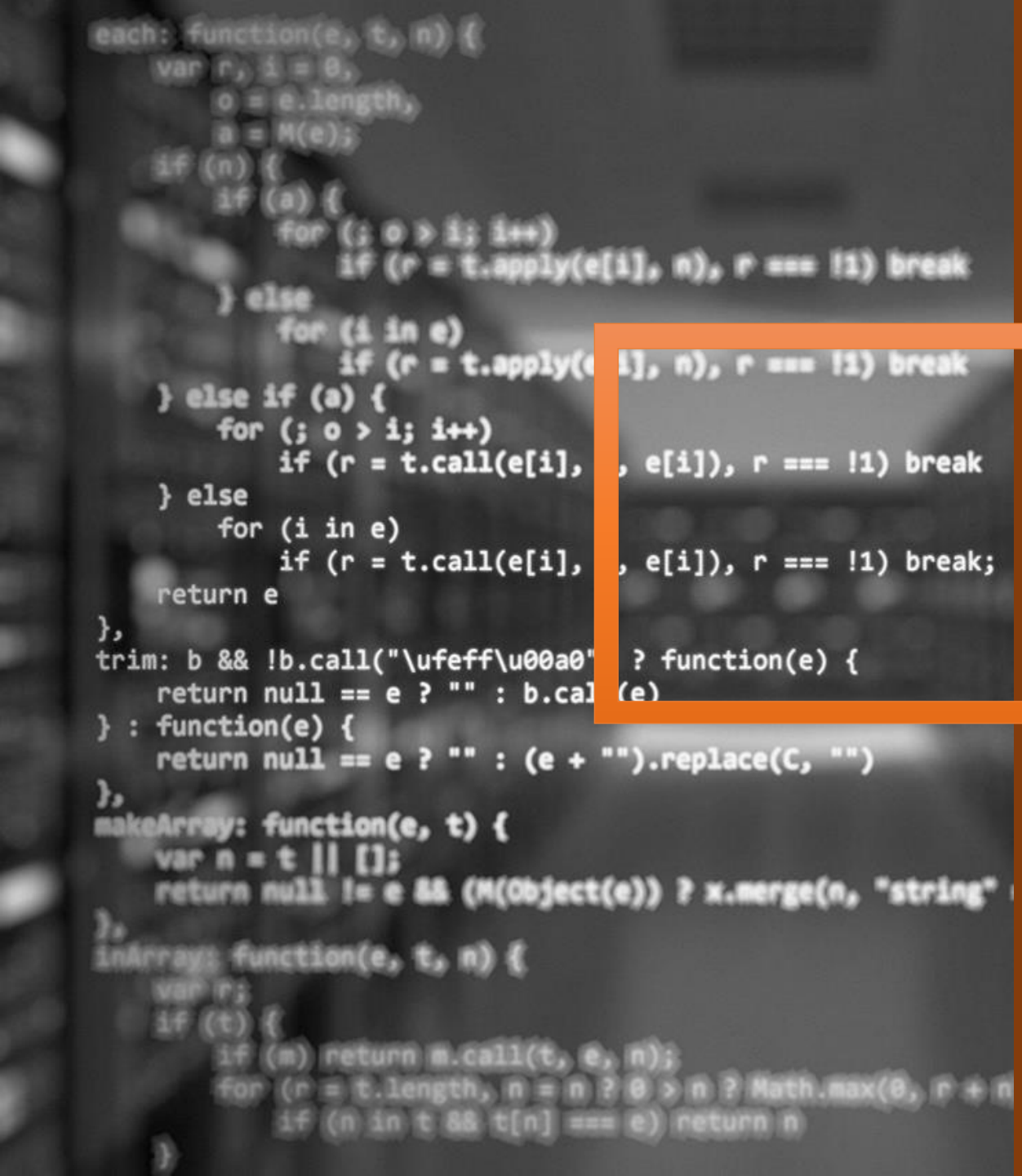
옵션	내용
-f, --force	강제로 삭제
--no-prune	부모가 untagged인 중간 이미지는 삭제하지 않음

docker image

- docker image tag 이미지명 별명
  - 이미지에 새로운 별명을 부여하는 명령어 (새로운 이미지가 생기는 것은 아님)

```
mgt@server2:~$ docker image tag mysql:8.0 mysql-image:8.0
mgt@server2:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
mysql-image   8.0       6c55ddbef969   3 months ago    591MB
mysql         8.0       6c55ddbef969   3 months ago    591MB
```

원본과 별명의 이미지ID가 동일하므로 둘은 같은 이미지임



03. 도커 컨테이너 명령어

docker container

- docker container 하위명령 [옵션]

하위명령	내용
create	이미지를 기반으로 컨테이너를 생성하는 명령어
start	생성 또는 중단된 컨테이너를 실행하는 명령어
stop	실행 중인 컨테이너를 중지하는 명령어
run	이미지를 내려 받아 컨테이너를 생성하고 실행하는 명령어 (image pull + container create / start)
rm	생성된 컨테이너를 삭제하는 명령어
pause / unpause	실행 중인 컨테이너를 일시정지 / 재실행하는 명령어
ls	생성된 컨테이너들의 목록을 조회하는 명령어
attach	가동 중인 컨테이너 내부에 접근하는 명령어 (기존 컨테이너가 /bin/bash를 실행하고 있어야 사용할 수 있음)
exec	가동 중인 컨테이너 내부에 접근하는 명령어 (모든 컨테이너 내부에서 /bin/bash를 실행할 수 있음)
cp	호스트와 컨테이너 사이에 파일을 복사하는 명령어

docker container

- docker container run [옵션] 이미지명
  - 이미지를 내려 받고, 컨테이너를 생성하고, 컨테이너를 실행하는 명령어
  - 이미 내려 받은 이미지는 새로 내려 받지 않음
  - docker image pull + docker container create + docker container run 명령을 한 번에 수행

옵션	내용
--name	컨테이너의 이름 지정
--network	컨테이너를 지정한 도커 네트워크에 연결
-v, --volume	볼륨 설정 (바인드 마운트, 볼륨 마운트)
-d, --detach	컨테이너를 백그라운드에서 실행
--rm	컨테이너 종료 시 해당 컨테이너 삭제
--restart	컨테이너 중지 이후 동작 결정
-p, --publish	호스트와 컨테이너 포트 연결
-e, --env	컨테이너로 환경 변수 전달

docker container

- docker container start 컨테이너명
  - 생성된 컨테이너를 실행하는 명령어
  - 컨테이너의 STATUS가 Exited에서 Up으로 변경됨

- docker container stop 컨테이너명
  - 실행 중인 컨테이너를 중단하는 명령어
  - 컨테이너의 STATUS가 Up에서 Exited로 변경됨
  - 모든 컨테이너 중지 명령 (모든 컨테이너의 ID를 한 번에 전달하는 방식)
    - $ docker container stop $(docker ps -aq)

docker container

- docker container rm 컨테이너명
  - 중단된 컨테이너를 삭제하는 명령어
  - 컨테이너가 실행 중인 경우에는 삭제할 수 없음
  - 모든 컨테이너 삭제 명령 (모든 컨테이너의 ID를 한 번에 전달하는 방식)
    - $ docker container rm $(docker ps -aq)

docker container

- docker container ls [옵션] / docker ps [옵션]
  - 생성된 컨테이너 목록을 조회하는 명령어

옵션	내용
-a, --all	모든 컨테이너 조회 (중지되거나 실행 중인 컨테이너 모두 조회)
-f, --filter	지정된 key 값으로 조회
-q, --quiet	컨테이너의 ID만 조회
-s, --size	컨테이너 파일의 크기를 조회
--format	원하는 형식으로 조회
--no-trunc	각 결과 항목을 자르지 않고 모두 출력
-l, --lastest	최신 컨테이너 1개만 조회
-n, --last	최신 컨테이너 n개만 조회

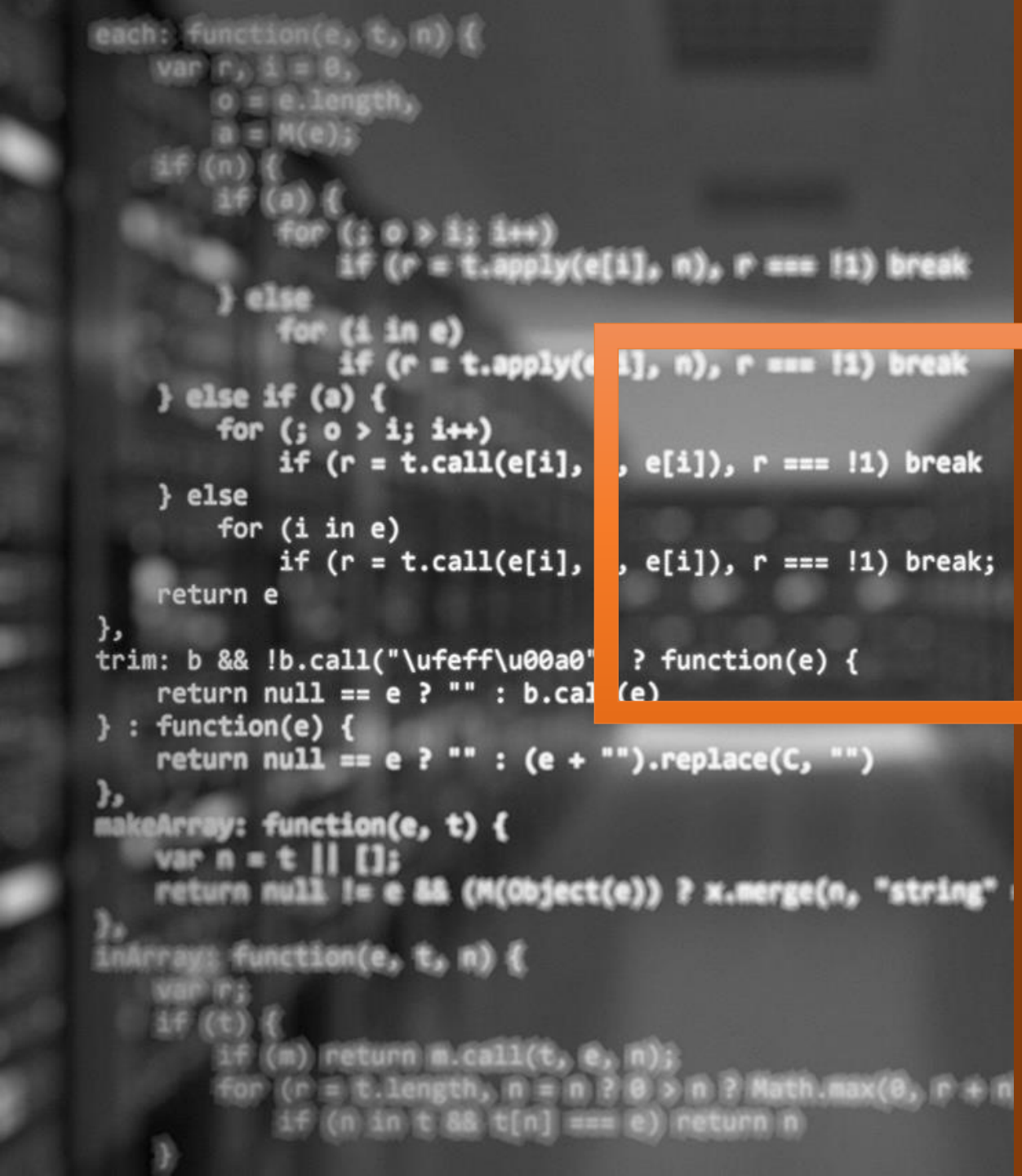
docker container

- docker container exec [옵션] 컨테이너 명령 [인자]
  - 실행 중인 컨테이너 내부에서 지정한 명령을 실행하는 명령어

옵션	내용
-i, --interactive	container에 표준 입력 열기
-t, --tty	tty(단말기 디바이스) 사용
-d, --detach	백그라운드 실행
-w, --workdir	컨테이너의 워킹 디렉터리 설정

docker container

- docker container cp [옵션] SRC_PATH DEST_PATH
  - 호스트 컴퓨터와 컨테이너 간 있는 파일이나 디렉터리를 복사하는 명령어
  - 컨테이너 경로를 작성할 땐 "컨테이너명:경로" 방식을 사용
  - 예시)
  - docker container cp ~/app.war tomcat-container:/webapps
    - 홈 디렉터리의 app.war 파일을 tomcat-container 컨테이너 내부의 /webapps 디렉터리에 복사
  - docker container cp ubuntu-container:/home/app.war ~/
    - ubuntu-container 컨테이너 내부의 /home/app.war 파일을 호스트 컴퓨터의 홈 디렉터리로 복사



04. 도커 볼륨 명령어

docker volume

- 도커 볼륨이란?
  - 컨테이너에 저장된 데이터는 해당 컨테이너가 삭제되면 함께 삭제됨
  - 컨테이너를 삭제한 이후에도 사용하던 데이터를 유지하고자 한다면 도커 볼륨 기능을 활용해야 함
    - 데이터베이스 구축을 컨테이너로 한 경우
    - 첨부 파일 등의 보관을 컨테이너로 한 경우
  - 볼륨 마운트와 바인딩 마운트 방식을 지원함
    - 볼륨 마운트 : 도커 볼륨을 만들어 컨테이너 특정 경로와 연결하는 방식
    - 바인딩 마운트 : 호스트의 디렉터리를 직접 컨테이너 특정 경로와 연결하는 방식
  - 도커 컨테이너를 실행할 때 --volume 또는 -v 옵션으로 볼륨 설정을 할 수 있음

docker volume

- docker volume 하위명령 [옵션]

하위명령	내용
create	도커 볼륨을 생성하는 명령어
ls	생성된 도커 볼륨들의 목록을 조회하는 명령어
inspect	특정 도커 볼륨의 상세 정보를 조회하는 명령어
prune	사용하지 않는 도커 볼륨을 삭제하는 명령어
rm	도커 볼륨을 삭제하는 명령어

Volume Mount

볼륨 마운트

- 도커 볼륨을 만들어서 컨테이너 내부 경로와 연결하는 방식
- 도커가 관리하기 때문에 바인딩 마운트 방식에 비해 안정적인 방식
- 도커에서 추천하는 방식
- 컨테이너 내부에 저장된 데이터가 자주 변하지 않는 경우에 적당함

볼륨 마운트 예시

- $ docker volume create some-volume
  - ➢ some-volume 이름을 가진 도커 볼륨 생성
- $ docker run -v some-volume:/home
  - ➢ 컨테이너 /home 경로의 모든 데이터는 컨테이너가 삭제되더라도 유지됨

Binding Mount

- 바인딩 마운트
  - 호스트의 특정 경로를 컨테이너 내부 경로와 직접 연결하는 방식
  - 호스트 경로에 쉽게 접근할 수 있으므로 컨테이너 내부에 자주 접근해야 하는 경우에 유리함

- 바인딩 마운트 예시
  - $ docker run -v /home:/tomcat
    - 호스트 /home 경로와 컨테이너 /tomcat 경로를 연결함
    - 컨테이너가 삭제되더라도 /tomcat 경로의 데이터는 호스트 /home 에서 확인할 수 있음