

docker-compose create helper script (dialog)

[pknw1logo-white.png](#)

```
/usr/local/bin/create-compose.sh
```

Script Headline

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.

```
# Create a temporary file to store dialog output
tempfile=$(mktemp /tmp/dialog.XXXXXX)
BACKGROUND_TITLE="Docker Compose Configurator"
dir=$(dirname $(pwd) | awk -F/ '{print $NF}')
existing_networks=$(docker network ls |awk '{print $2" "$3}'|grep -v NAME)
existing_users=$(cat /etc/passwd|egrep -v 'nologin|false'| awk -F: '{print $1" "$3}')
existing_user_ids=$(cat /etc/passwd|egrep -v 'nologin|false'| awk -F: '{print ":"$3":"}')
tz=$(cat /etc/timezone)


# Function to get user input with dialog
exec 3>&1;

function status() {
    status_update="${1}"
    echo "${status_update}" >> /tmp/dialog-log
}
```

```

function wrapper() {
    feature="${1}"
    title="${2}"
    dimensions="${3}"
    data "${4}"

    response=$(dialog --backtitle "${BACKGROUND_TITLE} - ${title}" \
        --${feature} "${title}" \
        "${dimensions}" \
        "${data}" \
        2>&1 1>&3 )

    exit_code=$?

    case $exit_code in
        0) status "${2} - success exit code"
            ;;
        3) status "${2} - alternate button"
            ;;
        *) status "${2} - non-zero error code"
            ;;
    esac

    if [[ -z "${response}" ]]
    then
        echo "${response}"
    fi
}

## SERVICE_NAME
    service_name=$(dialog --backtitle "${BACKGROUND_TITLE} - Service Name" --inputbox
'Service Name' 0 0 ${dir} 2>&1 1>&3);

## DOCKER COMPOSE IMAGE and DOCKER IMAGE TEST PULL
    image_name=$(dialog --backtitle "${BACKGROUND_TITLE} - Docker Image Name" --inputbox
'Image Name' 0 0 2>&1 1>&3);

    rm /tmp/image_pull
    image_test=$(docker pull $image_name || echo error > /tmp/image_pull)
    if [[ $(cat /tmp/image_pull) == "error" ]]

```

```

then
    image_name=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox
'Docker Image not found - Try again' 0 0 2>&1 1>&3);
fi

## TOP LEVEL DOMAIN

top_level_domain=$(dialog --backtitle "${BACKGROUND_TITLE}" --title "Top Level Domain" --
inputbox "domain" 8 50 "pknw1.co.uk" 2>&1 1>&3);

## DOCKER CONFIG ROOT FOLDER

config_folder=$(dialog --backtitle "${BACKGROUND_TITLE}" --title "config folder" --
inputbox "enter the docker config root path" 8 50 "/etc/docker/config" 2>&1 1>&3)

## DOCKER NETWORK SELECTION & CREATION

docker_network=$(dialog --backtitle "${BACKGROUND_TITLE}" --menu "Choose Network: " 22 50
15 ${existing_networks} New Network 2>&1 1>&3)
if [[ ${docker_network} == "New" ]]
then
    docker_network=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox 'New Network
Name:' 0 0 2>&1 1>&3);
fi

## USER SELECTION

choose_user=$(dialog --backtitle "${BACKGROUND_TITLE}" --menu "Choose User: " 22 35 10
${existing_users} 2>&1 1>&3)
rm /tmp/id

## GROUP SELECTION

for ID in ${existing_user_ids}; do cat /etc/group | grep $ID|awk -F: '{print $1" "$3}' >>
/tmp/id; done
existing_groups=$(cat /tmp/id| sort -u)
select_group=$(dialog --backtitle "${BACKGROUND_TITLE}" --menu "Choose Group: " 22 35 10
${existing_groups} 2>&1 1>&3)

## RESTART POLICY

restart=$(dialog --backtitle "${BACKGROUND_TITLE}" --menu "Choose Restart Option: " 22 35 10
no 'never restart' always 'always start' on-failure 'fail' unless-stopped 'unl' 2>&1 1>&3)

## NGINX-PROXY APP PORT

app_port=$(dialog --inputbox 'App Port for Proxy' 0 0 "8080" 2>&1 1>&3);

```

```

### Volumes and Mounts

active_volumes=$(docker ps --format "{{.Names}}" | xargs -n1 docker inspect -f '{{json
.Mounts}}' | jq -r '.[ ] | "\(.Source):\(.Destination) \(.Source):\(.Destination) off"' |sort -
u)

filtered_volumes=$(echo "${active_volumes}" | egrep -v 'sock|squid|yaml|config' | sed
's/media off/media on/g'| sed 's/fuse off/fuse on/g')

selected_volumes=$(dialog --backtitle "${BACKGROUND_TITLE} - Add predefined volumes" --
extra-button --extra-label "Add Custom Mounts" --checklist "Select Volumes to Map" 20 100 20
$filtered_volumes 2>&1 1>&3);

dialog_exit_code=$?

case $dialog_exit_code in
0) echo "Add Selected Volumes"
    echo ${selected_volumes}
    add_volume="false"
    ;;
1) echo "Cancel"
    add_volume="false"
    ;;
3) echo "Add Custom Mount"
    add_volume="true"
    ;;
*) echo ""
    add_volume="false"
    ;;
esac

volumes=()

for SEL in "${selected_volumes}"
do
    volumes+=("${SEL}")
done

while [[ "${add_volume}" == "true" ]]
do
    vol=$(dialog --backtitle "${BACKGROUND_TITLE} - Add Custom Mount" --separate-widget
$'\n' --title "Add custom mount" --extra-button --extra-label "Add Another" --form "" 0 0 0
"Source:" 1 1 "$source" 1 10 30 0 "Mount:" 2 1 "$mount" 2 10 30 0 2>&1 1>&3 )

```

```

dialog_exit_code=$?
form=$(echo ${vol}| tr ' ' ':')
case $dialog_exit_code in
    0) echo "Add Single Custom"
        volumes+=" ${form}"
        add_volume="false"
        ;;
    1) echo "Cancel"
        add_volume="false"
        ;;
    3) echo "Add More"
        volumes+=" ${form}"
        add_volume="true"
        ;;
    *) echo ""
        add_volume="false"
        ;;
esac
done

## PRIVILEGED MODE
if dialog --clear --backtitle "${BACKGROUND_TITLE}" --title "Privileged Mode" --yesno
"Enable Privileged Mode?" 0 0 ;then priv=true;else priv=false;fi

## DNS OVERRIDE
dns=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox 'Override Primary DNS' 0 0
"8.8.8.8" 2>&1 1>&3);

## RESOURCE LIMITING
cpus=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox 'Resource Allocate CPUs' 0 0
"1" 2>&1 1>&3);
memory=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox 'Resource Allocate Memory' 0
0 "200" 2>&1 1>&3);

## TimeZone
timezone=$(dialog --backtitle "${BACKGROUND_TITLE}" --inputbox 'Select Timezone' 0 0 $tz
2>&1 1>&3);

exec 3>&-;

```

```
rm -f "$tempfile"
```

```
# Display results to confirm input
```

```
dialog --backtitle "${BACKGROUND_TITLE} - Configuration Summary" --title "Confirmation" --
```

```
msgbox "\
```

```
    Service Name: $service_name\n\
```

```
    Image Name: $image_name\n\
```

```
    Top Level Domain: $top_level_domain\n\
```

```
    Config Folder: $config_folder\n\
```

```
    Network: $docker_network\n\
```

```
    User: $choose_user\n\
```

```
    Group: $select_group\n\
```

```
    Privileged: $priv\n\
```

```
    Vhost: "${service_name}.pknw1.co.uk"\n\
```

```
    Port: $app_port\n\
```

```
    DNS: $dns\n\
```

```
    CPU: $cpus\n\
```

```
    Memory: $memory\n\
```

```
    TimeZone: $timezone\n\
```

```
    Volumes: $volumes\n\
```

```
    Timezone: $timezone" 20 70
```

```
clear
```

sdfsd

☐ script info

☐ check mark

further info

Product Home

[Link](#)

<u>Documentation</u>	Link
<u>Github</u>	Link
<u>DockerHub</u>	Link
<u>Misc</u>	Link

more text more text

Revision #1
Created 7 January 2025 09:19:16 by pknw1
Updated 7 January 2025 09:19:17 by pknw1