Red Hat Ansible Automation Platform 2.6- Single Node Installation (Step-by-Step)

Overview

This guide covers installing all AAP components on a **single server** for testing, development, or small production environments.

Phase 1: System Preparation

Step 1: Verify System Requirements

```
# Check RHEL version
cat /etc/redhat-release
# Required: Red Hat Enterprise Linux release 9.4 or later

# Check hostname
hostname
hostnamectl

# Check system resources
echo "=== CPU Check ==="
lscpu | grep "^CPU(s):"
# Minimum: 4 vCPUs

echo "=== Memory Check ==="
free-h
# Minimum: 16GB RAM

echo "=== Disk Space Check ==="
df-h
# Minimum: 80GB available
```

Step 2: Set Hostname (if not already set)

```
# Set your hostname
sudo hostnamectl set-hostname aap.example.com
```

```
# Update /etc/hosts
sudo bash-c 'cat >> /etc/hosts << EOF
127.0.0.1   aap.example.com aap
EOF'

# Verify
hostname
hostname-f
```

Step 3: Update System

```
# Update all packages
sudo dnf update-y

# Reboot if kernel updated
sudo reboot
```

Step 4: Install and Configure NTP

```
# Install chrony
sudo dnf install chrony-y

# Start and enable service
sudo systemctl start chronyd
sudo systemctl enable chronyd

# Verify NTP sync
sudo chronyc tracking
```

Step 5: Configure Firewall

```
# Check firewall status
sudo systemctl status firewalld

# If not running, start it
sudo systemctl start firewalld
sudo systemctl enable firewalld
```

Open required ports

sudo firewall-cmd--permanent--add-service=http

sudo firewall-cmd--permanent--add-service=https

sudo firewall-cmd--permanent--add-port=5432/tcp # PostgreSQL

sudo firewall-cmd--permanent--add-port=27199/tcp # Receptor

sudo firewall-cmd--permanent--add-port=8080/tcp # Gateway HTTP

sudo firewall-cmd--permanent--add-port=8443/tcp # Gateway HTTPS

Reload firewall

sudo firewall-cmd--reload

Verify

sudo firewall-cmd--list-all

Step 6: Verify SELinux

Check SELinux status

getenforce

Should show: Enforcing

If not enforcing, enable it

sudo setenforce 1

Make persistent

sudo sed-i 's/SELINUX=permissive/SELINUX=enforcing/' /etc/selinux/config

sudo sed-i 's/SELINUX=disabled/SELINUX=enforcing/' /etc/selinux/config

---

Phase 2: Red Hat Subscription

Step 7: Register System with Red Hat

Register with Red Hat Subscription Manager

sudo subscription-manager register--username YOUR_RH_USERNAME--password YOUR_RH_PASSWORD

You'll see output like:

The system has been registered with ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Step 8: Verify Registration

Refresh subscription

sudo subscription-manager refresh

Check registration status

sudo subscription-manager identity

Expected output:

system identity: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

name: aap.example.com

org name: Your Organization

org ID: 1234567

Step 9: Enable Required Repositories

Enable BaseOS repository

sudo subscription-manager repos--enable rhel-9-for-x86_64-baseos-rpms

Enable AppStream repository

sudo subscription-manager repos--enable rhel-9-for-x86_64-appstream-rpms

Verify enabled repositories

sudo subscription-manager repos--list-enabled

Clean and update cache

sudo dnf clean all

sudo dnf makecache

Phase 3: Registry Service Account

Step 10: Create Red Hat Registry Service Account

Open browser and navigate to:

https://access.redhat.com/terms-based-registry/accounts

Steps:

1. Click "New Service Account"
2. Name: ansible-automation-platform-prod
3. Description: Service account for AAP 2.6 installation
4. Click "Create"
5. Copy and save securely:
    - Username: (looks like 12345678|ansible-automation-platform-prod)
    - Token: (long alphanumeric string)

Save these credentials- you'll need them in Step 13!

Phase 4: Download and Extract Installer

Step 11: Download Installer

Option A: From Red Hat Customer Portal (recommended)

1. Navigate to: https://access.redhat.com/downloads/
2. Select: Red Hat Ansible Automation Platform
3. Version: 2.6
4. Download: "Ansible Automation Platform 2.6 Setup Bundle"
5. File: ansible-automation-platform-setup-bundle-2.6-1.tar.gz

Option B: Using wget (if you have download URL)

```
# Create download directory
mkdir-p ~/downloads
cd ~/downloads

# Download (replace URL with actual download link)
wget https://access.cdn.redhat.com/.../ansible-automation-platform-setup-bundle-2.6-1.tar.gz
```

Step 12: Extract Installer

```
# Create installation directory
sudo mkdir-p /opt/ansible-automation-platform
```

Copy downloaded file

sudo cp ~/downloads/ansible-automation-platform-setup-bundle-2.6-1.tar.gz /opt/ansible-automation-platform/

Change to installation directory

cd /opt/ansible-automation-platform

Extract

sudo tar xvf ansible-automation-platform-setup-bundle-2.6-1.tar.gz

Change to extracted directory

cd ansible-automation-platform-setup-bundle-2.6-1

List contents

ls-la

You should see:

drwxr-xr-x collections/

drwxr-xr-x group_vars/

-rw-r--r-- inventory

-rwxr-xr-x setup.sh

...

Phase 5: Configure Installation

Step 13: Backup Original Inventory

Still in: /opt/ansible-automation-platform/ansible-automation-platform-setup-bundle-2.6-1

Backup original inventory

sudo cp inventory inventory.original

Step 14: Create Single Node Inventory File

Edit inventory file

sudo vi inventory

Press i to enter INSERT mode, then paste this configuration:

```
# Single Node Ansible Automation Platform Installation
# All components on one server

[automationcontroller]
aap.example.com

[automationgateway]
aap.example.com

[database]
aap.example.com

[all:vars]
# ============================================
# ADMIN CREDENTIALS
# ============================================
# Admin password for Automation Controller
admin_password='MySecurePass123!'

# ============================================
# REDIS CONFIGURATION
# ============================================
redis_mode=standalone

# ============================================
# POSTGRESQL DATABASE
# ============================================
pg_host='aap.example.com'
pg_port=5432
pg_database='awx'
```

```
pg_username='awx'
pg_password='AwxDbPass123!'
pg_sslmode='prefer'

# ============================================
# REGISTRY CREDENTIALS
# ============================================
# Replace with YOUR credentials from Step 10
registry_url='registry.redhat.io'
registry_username='12345678|your-service-account-name'
registry_password='your-very-long-token-string-here'

# ============================================
# AUTOMATION GATEWAY
# ============================================
automationgateway_admin_password='GatewayPass123!'
automationgateway_pg_host='aap.example.com'
automationgateway_pg_port=5432
automationgateway_pg_database='automationgateway'
automationgateway_pg_username='automationgateway'
automationgateway_pg_password='GatewayDbPass123!'
automationgateway_pg_sslmode='prefer'

# ============================================
# SSL/TLS CONFIGURATION (Optional)
# ============================================
# Uncomment if using custom certificates
# custom_ca_cert=/path/to/ca.crt
# web_server_ssl_cert=/path/to/tower.cert
# web_server_ssl_key=/path/to/tower.key
# automationgateway_ssl_cert=/path/to/gateway.cert
```

automationgateway_ssl_key=/path/to/gateway.key

Important: Replace these values with your own:

- aap.example.com → Your actual hostname
- MySecurePass123! → Your strong admin password
- AwxDbPass123! → Your database password
- GatewayPass123! → Your gateway admin password
- GatewayDbPass123! → Your gateway database password
- registry_username → From Step 10
- registry_password → Token from Step 10

Save and exit:

- Press ESC
- Type :wq
- Press ENTER

Step 15: Verify Inventory Configuration

Check for syntax errors

sudo cat inventory | grep-E 'password|username' | head-10

Verify hostname resolution

ping-c 2 aap.example.com

Verify localhost resolution

ping-c 2 localhost

Phase 6: Run Installation

Step 16: Set Proper Permissions

Set umask

umask 0022

Make setup script executable

sudo chmod +x setup.sh

Verify

ls-la setup.sh

Step 17: Run Pre-Installation Check

Check if Ansible is available

which ansible-playbook

Check Python version

python3--version

Step 18: Start Installation

Run installation (this will take 20-45 minutes)

sudo ./setup.sh

What happens during installation:

1. ✓ Verifying Ansible installation
2. ✓ Installing required packages
3. ✓ Configuring PostgreSQL database
4. ✓ Creating database users and schemas
5. ✓ Installing Automation Controller
6. ✓ Installing Automation Gateway
7. ✓ Configuring Redis
8. ✓ Installing Receptor
9. ✓ Configuring NGINX
10. ✓ Starting services
11. ✓ Running database migrations
12. ✓ Creating admin user

Monitor the output for errors. Successful installation ends with:

PLAY RECAP **********

aap.example.com : ok=XXX changed=XXX unreachable=0 failed=0

The setup process completed successfully.

Setup log saved to /var/log/tower/setup-YYYY-MM-DD-HH:MM:SS.log

Step 19: Verify Installation Completed

```
# Check if installation log shows success
sudo tail-20 /var/log/tower/setup-*.log

# Look for "The setup process completed successfully"
```

Phase 7: Verify Services

Step 20: Check All Services

```
# Check Automation Controller
sudo systemctl status automation-controller

# Check Receptor
sudo systemctl status receptor

# Check PostgreSQL
sudo systemctl status postgresql

# Check NGINX
sudo systemctl status nginx

# Check Redis
sudo systemctl status redis

# List all AAP-related services
sudo systemctl list-units--type=service--state=running | grep-E
'automation|receptor|nginx|postgresql|redis'
```

All services should show: active (running)

Step 21: Check Service Logs

```
# Controller logs
sudo tail-50 /var/log/tower/tower.log
```

NGINX logs

sudo tail-20 /var/log/nginx/error.log

PostgreSQL logs

sudo ls-la /var/lib/pgsql/data/log/

sudo tail-20 /var/lib/pgsql/data/log/postgresql-*.log

Step 22: Verify Ports Are Listening

Check listening ports

sudo ss-tlnp | grep-E ':80|:443|:5432|:27199|:8080|:8443'

Expected output should show:

:80 - NGINX HTTP

:443 - NGINX HTTPS

:5432 - PostgreSQL

:8080 - Gateway HTTP

:8443 - Gateway HTTPS

:27199- Receptor

Step 23: Test HTTP Connectivity

Test localhost

curl-k http://localhost

Test HTTPS

curl-k https://localhost

Test gateway

curl-k https://localhost:8443

Test with hostname

curl-k https://aap.example.com

Phase 8: Obtain and Upload Subscription

Step 24: Create Subscription Manifest

Open browser and navigate to:

https://access.redhat.com/management/subscription_allocations

Create Allocation:

1. Click "New Subscription Allocation"
2. Name: Ansible Automation Platform 2.6
3. Type: Select "Satellite 6.16"
4. Click "Create"

Add Subscriptions:

1. Click on your allocation name
2. Click "Subscriptions" tab
3. Click "Add Subscriptions"
4. Find: "Red Hat Ansible Automation Platform"
5. Quantity: Enter number of nodes (minimum 1)
6. Click "Submit"

Download Manifest:

1. Click "Export Manifest"
2. Save file: manifest_ansible_automation_platform.zip

Step 25: Access Web UI

Open browser:

https://aap.example.com

Or use IP address:

https://YOUR_SERVER_IP

Accept SSL Warning:

- Click "Advanced"
- Click "Accept the Risk and Continue" (or similar)

Login Screen Appears:

- Username: admin
- Password: (your admin_password from inventory)

Step 26: Upload Subscription Manifest

Subscription Wizard appears automatically:

1. Click **"Subscription manifest"** tab
2. Click **"Browse"** button
3. Select downloaded manifest file
4. Check box: **"I agree to the End User License Agreement"**
5. Click **"Finish"**

Wait for upload to complete (30-60 seconds)

Verify Success:

- Status: **Compliant** (green checkmark)
- **Hosts Automated:** Shows available count
- **Subscription Valid:** Shows expiration date

Step 27: Configure Automation Analytics

Default: Opted IN to share anonymous usage data

To opt out:

1. Click hamburger menu (☰)
2. Navigate: **Settings → Automation Execution → System**
3. Scroll to: **Gather data for Automation Analytics**
4. Uncheck the box
5. Click **"Save"**

Phase 9: Initial Configuration

Step 28: Explore the Dashboard

Main sections:

- **Dashboard:** Overview of automation activity
- **Automation Execution:** Job templates, projects, inventories
- **Automation Decisions:** Event-Driven Ansible (if installed)
- **Access Management:** Users, teams, organizations
- **Administration:** Settings, subscription, tasks

Step 29: Create Your First Organization

1. Click **"Access Management"** → **"Organizations"**
2. Click **"Create organization"** button
3. Configure:

- **Name:** My Organization
- **Description:** Default organization for automation

4. Click **"Create organization"**

Step 30: Create Additional User

1. Click **"Access Management"** → **"Users"**
2. Click **"Create user"**
3. Configure:
   - **Username:** automation-admin
   - **Email:** admin@example.com
   - **Password:** (set strong password)
   - **Confirm Password:** (repeat password)
   - **User Type:** System Administrator
4. Click **"Create user"**

Step 31: Create Credential for SSH Access

1. Click **"Automation Execution"** → **"Infrastructure"** → **"Credentials"**
2. Click **"Create credential"**
3. Configure:
   - **Name:** Demo SSH Credential
   - **Organization:** My Organization
   - **Credential Type:** Machine
   - **Username:** root (or your SSH user)
   - **SSH Private Key:**
     - Click **"Browse"** to upload
     - Or paste key directly
4. Click **"Create credential"**

Step 32: Create Demo Project

1. Click **"Automation Execution"** → **"Projects"**
2. Click **"Create project"**
3. Configure:
   - **Name:** Demo Project
   - **Organization:** My Organization

- **Source Control Type:** Git
- **Source Control URL:** https://github.com/ansible/ansible-examples.git
- Leave other defaults

4. Click **"Create project"**

Wait for sync:

- Status changes from "Pending" to "Successful"
- Takes 10-30 seconds

Step 33: Create Inventory

1. Click **"Automation Execution"** → **"Infrastructure"** → **"Inventories"**
2. Click **"Create inventory"** → **"Create inventory"**
3. Configure:
   - **Name:** Demo Inventory
   - **Organization:** My Organization
4. Click **"Create inventory"**

Add Host:

1. Click on **"Demo Inventory"**
2. Click **"Hosts"** tab
3. Click **"Create host"**
4. Configure:
   - **Name:** localhost
   - **Variables:** (optional)
5. ansible_connection: local
6. Click **"Create host"**

Phase 10: Test Your Installation

Step 34: Create Simple Test Playbook Project

Create a test Git repository or use existing:

For quick test, we'll use localhost:

1. Click **"Automation Execution"** → **"Templates"**
2. Click **"Create template"** → **"Create job template"**
3. Configure:

- **Name:** Hello World Test
- **Job Type:** Run
- **Inventory:** Demo Inventory
- **Project:** Demo Project
- **Playbook:** Select one from dropdown (e.g., helloworld.yml)
- **Credentials:** Demo SSH Credential
- **Execution Environment:** Default execution environment

4. Click **"Create job template"**

Step 35: Run Your First Job

1. On the **Hello World Test** template, click **"Launch"** button
2. **Job starts running:**
   - Watch real-time output
   - See each task execute
   - View results

Successful job shows:

- Status: **Successful** (green)
- All tasks: **ok**
- No failures

Step 36: View Job History

1. Click **"Views"** → **"Jobs"**
2. See your completed job
3. Click on job to view details:
   - Full output
   - Timing
   - Host facts
   - Events

Phase 11: Backup and Documentation

Step 37: Create Initial Backup

```
# Navigate to installer directory
cd /opt/ansible-automation-platform/ansible-automation-platform-setup-bundle-2.6-1
```

```
# Create backup directory
sudo mkdir-p /var/backups/ansible

# Run backup
sudo ./setup.sh-e 'backup_dest=/var/backups/ansible' \
 -e 'use_archive_compression=true' \
 -e 'use_db_compression=true'-b

# Wait for backup to complete (5-10 minutes)

# Verify backup
ls-lh /var/backups/ansible/
```

You should see:

automation-platform-backup-YYYY-MM-DD-HH-MM-SS.tar.gz

Step 38: Document Your Installation

```
# Create documentation file
sudo tee /opt/ansible-automation-platform/INSTALLATION_INFO.txt << EOF
=====================================
Ansible Automation Platform 2.6
Single Node Installation
=====================================

Installation Date: $(date)
Hostname: $(hostname-f)
IP Address: $(hostname-I | awk '{print $1}')
RHEL Version: $(cat /etc/redhat-release)

=====================================
COMPONENTS INSTALLED
=====================================
```

- Automation Controller

- Automation Gateway

- PostgreSQL Database

- Redis Cache

- Receptor

- NGINX Web Server

==========

ACCESS INFORMATION

==========

Web UI: https://$(hostname-f)

Admin User: admin

Admin Password: [STORED SECURELY]

==========

DATABASE INFORMATION

==========

PostgreSQL Host: localhost

Controller DB: awx

Gateway DB: automationgateway

PostgreSQL Port: 5432

==========

BACKUP INFORMATION

==========

Backup Location: /var/backups/ansible/

Latest Backup: $(ls-t /var/backups/ansible/*.tar.gz 2>/dev/null | head-1)

==========

INSTALLATION LOG

==========

```
$(ls-t /var/log/tower/setup-*.log | head-1)

====================================
NOTES
====================================
- Firewall configured for HTTP/HTTPS access
- SELinux in enforcing mode
- Self-signed SSL certificates in use
- Registry: registry.redhat.io

Installation completed by: $(whoami)
====================================
EOF

# Display documentation
cat /opt/ansible-automation-platform/INSTALLATION_INFO.txt
```

Step 39: Save Important Files

```
# Create configuration backup directory
sudo mkdir-p /root/aap-config-backup

# Backup inventory file
sudo cp /opt/ansible-automation-platform/ansible-automation-platform-setup-bundle-2.6-1/inventory
\
  /root/aap-config-backup/inventory.$(date +%Y%m%d)

# Backup installation info
sudo cp /opt/ansible-automation-platform/INSTALLATION_INFO.txt \
  /root/aap-config-backup/

# Create archive of important files
cd /root
```

```
sudo tar czf aap-config-$(date +%Y%m%d).tar.gz aap-config-backup/

# List backup
ls-lh /root/aap-config-*.tar.gz
```

Phase 12: Health Check and Verification

Step 40: System Health Check

```
# Create health check script
sudo tee /usr/local/bin/aap-health-check.sh << 'EOF'
#!/bin/bash
echo "========================================="
echo "Ansible Automation Platform Health Check"
echo "========================================="
echo "Date: $(date)"
echo ""

echo "=== Service Status ==="
services=("automation-controller" "receptor" "postgresql" "nginx" "redis")
for service in "${services[@]}"; do
    if systemctl is-active--quiet $service; then
        echo "✓ $service: Running"
    else
        echo "✗ $service: Not Running"
    fi
done
echo ""

echo "=== Port Status ==="
ports=("80:HTTP" "443:HTTPS" "5432:PostgreSQL" "8080:Gateway" "8443:Gateway-HTTPS")
for port_info in "${ports[@]}"; do
    port="${port_info%%:*}"
```

```
    name="${port_info##*:}"
    if ss-tln | grep-q ":$port "; then
        echo "✓ $name (Port $port): Listening"
    else
        echo "✗ $name (Port $port): Not Listening"
    fi
done
echo ""

echo "=== Disk Usage ==="
df-h / /var | grep-v "Filesystem"
echo ""

echo "=== Memory Usage ==="
free-h | grep "Mem:"
echo ""

echo "=== System Load ==="
uptime
echo ""

echo "=== Recent Errors (last 10) ==="
journalctl-p err-n 10--no-pager 2>/dev/null || echo "No recent errors"
echo ""

echo "========================================"
echo "Health Check Complete"
echo "========================================"
EOF

# Make executable
```

```
sudo chmod +x /usr/local/bin/aap-health-check.sh

# Run health check
sudo /usr/local/bin/aap-health-check.sh
```

Step 41: Test API Access

```
# Test API ping endpoint
curl-k https://localhost/api/v2/ping/

# Expected response:
# {"ha":false,"version":"X.X.X","active_node":"aap.example.com"}

# Get API version
curl-k https://localhost/api/v2/

# Test with authentication
# First, get token (replace admin password)
TOKEN=$(curl-k-s-X POST \
 -H "Content-Type: application/json" \
 -d '{"username":"admin","password":"YOUR_ADMIN_PASSWORD"}' \
  https://localhost/api/gateway/v1/tokens/ | grep-o '"token":"[^"]*' | cut-d'"'-f4)

echo "Token: $TOKEN"

# Use token to access API
curl-k-H "Authorization: Bearer $TOKEN" \
  https://localhost/api/gateway/v1/ping/
```

Step 42: Final Verification Checklist

```
# Run comprehensive checks
echo "=== Final Installation Verification ==="
echo ""
```

```
# 1. Hostname resolution
echo "1. Hostname Resolution:"
hostname-f
getent hosts $(hostname-f)
echo ""

# 2. Services
echo "2. Critical Services:"
systemctl is-active automation-controller && echo "✓ Controller" || echo "✗ Controller"
systemctl is-active postgresql && echo "✓ Database" || echo "✗ Database"
systemctl is-active nginx && echo "✓ Web Server" || echo "✗ Web Server"
echo ""

# 3. Web UI
echo "3. Web UI Access:"
curl-k-s-o /dev/null-w "HTTP Status: %{http_code}\n" https://localhost/
echo ""

# 4. Database connectivity
echo "4. Database Connectivity:"
sudo-u awx psql-h localhost-U awx-d awx-c "SELECT 1;" 2>/dev/null && echo "✓ Database Connected" || echo "✗ Database Connection Failed"
echo ""

# 5. Disk space
echo "5. Disk Space:"
df-h / | awk 'NR==2 {print "Root: " $5 " used of " $2}'
df-h /var | awk 'NR==2 {print "Var: " $5 " used of " $2}'
echo ""

echo "=== Verification Complete ==="
```

Troubleshooting Guide

Issue: Services Not Starting

```
# Check service status
sudo systemctl status automation-controller-l

# View logs
sudo journalctl-u automation-controller-n 50

# Reset and restart
sudo systemctl reset-failed
sudo systemctl restart automation-controller
```

Issue: Cannot Access Web UI

```
# Check NGINX is running
sudo systemctl status nginx

# Check NGINX configuration
sudo nginx-t

# Check firewall
sudo firewall-cmd--list-all

# Check SELinux denials
sudo ausearch-m avc-ts recent
```

Issue: Database Connection Failed

```
# Check PostgreSQL is running
sudo systemctl status postgresql

# Check PostgreSQL is listening
sudo ss-tlnp | grep 5432
```

Test database connection

sudo-u postgres psql-l

Check database logs

sudo tail-50 /var/lib/pgsql/data/log/postgresql-*.log

Issue: Installation Failed

View installation log

sudo cat /var/log/tower/setup-*.log | less

Search for errors

sudo grep-i error /var/log/tower/setup-*.log

Check disk space

df-h

Check memory

free-h

Re-run installation

cd /opt/ansible-automation-platform/ansible-automation-platform-setup-bundle-2.6-1

sudo ./setup.sh

Installation Complete! ✅

What You Have Now:

✓ Fully functional Ansible Automation Platform

- Automation Controller (automation execution)
- Platform Gateway (unified authentication)
- PostgreSQL Database (data storage)
- Redis Cache (performance)
- Receptor (job distribution)

Access Your Installation:

Web UI: https://aap.example.com or https://YOUR_IP **Username:** admin **Password:** (from your inventory file)

Next Steps:

1. **Security:** Replace self-signed certificates with proper SSL certs
2. **Integration:** Configure LDAP/AD for user authentication
3. **Automation:** Import your playbooks and inventories
4. **Organization:** Create teams and assign permissions
5. **Monitoring:** Set up log forwarding and monitoring
6. **Backup:** Schedule regular backups

Useful Commands:

```
# Health check
sudo /usr/local/bin/aap-health-check.sh

# View logs
sudo tail-f /var/log/tower/tower.log

# Restart services
sudo systemctl restart automation-controller

# Create backup
cd /opt/ansible-automation-platform/ansible-automation-platform-setup-bundle-2.6-1
sudo ./setup.sh-b
```

Support Resources:

- **Documentation:** https://docs.ansible.com
- **Support Portal:** https://access.redhat.com/support
- **Community:** https://forum.ansible.com
- **Knowledge Base:** https://access.redhat.com/articles/

🎉 **Congratulations! Your single-node Ansible Automation Platform is ready to use!**