

ComfyUI Starter Workflow Pack

A Progressive Learning Path for Experimental Learners

Workflow 1: Basic Text-to-Image (START HERE)

Goal: Understand the fundamental pipeline

What It Does:

Prompt → Load Model → Generate Image → Save

Nodes You'll See:

- **Load Checkpoint** - Loads your model (SDXL, SD1.5, etc.)
- **CLIP Text Encode (Prompt)** - Your positive prompt
- **CLIP Text Encode (Prompt)** - Your negative prompt
- **KSampler** - The "brain" that generates the image
- **VAE Decode** - Converts latent space to visible image
- **Save Image** - Outputs the final result

Experiments to Try:

1. Change the prompt text - try "a cat" vs "a detailed portrait of a cat"
2. Adjust **steps** in KSampler (try 20, 30, 40) - more steps = more refined
3. Change **cfg** (guidance scale) - try 7, 10, 15 - higher = follows prompt more strictly
4. Try different **samplers** (euler, dpmpp_2m, etc.) - each has a different "style"
5. Change the **seed** - different number = different image
6. Set seed to -1 for random results each time

What to Notice:

- How does generation time change with steps?
 - Does higher CFG always look better?
 - Which sampler do you prefer?
-

Workflow 2: Adding LoRAs

Goal: Learn how to modify model behavior with smaller add-ons

New Nodes:

- **Load LoRA** - Adds style/character/concept to your base model

What Changes:

The LoRA node sits between your checkpoint and the CLIP/model inputs

Experiments to Try:

1. Load a style LoRA - try strength values from 0.3 to 1.5
2. Stack two LoRAs - see how they interact
3. Remove the LoRA entirely - compare before/after
4. Try the same prompt with different LoRAs
5. Use negative LoRA strength (like -0.5) - see what happens!

What to Notice:

- At what strength does the LoRA become too overpowering?
 - Do some LoRAs work better at low vs high strength?
 - How do multiple LoRAs affect generation time?
-

Workflow 3: Image-to-Image with ControlNet

Goal: Use a reference image to guide generation

New Nodes:

- **Load Image** - Imports your reference image
- **ControlNet Loader** - Loads a control model (depth, canny, pose, etc.)
- **Apply ControlNet** - Connects control to your generation

Experiments to Try:

1. Load a photo and use canny edge detection - try to recreate it in different styles
2. Adjust ControlNet **strength** (0.5 to 1.5) - how much control vs creativity?
3. Use the same control image with different prompts
4. Try different ControlNet types (depth, normal, lineart)
5. Turn ControlNet strength to 0 - what's the difference?

What to Notice:

- Which ControlNet type preserves structure best?
 - Can you completely override the control with a strong prompt?
 - How does this compare to img2img without ControlNet?
-

Workflow 4: Upscaling & Refinement

Goal: Take a small image and make it larger with quality

New Nodes:

- **Upscale Image** - Makes image bigger
- **Load Upscale Model** - Loads ESRGAN or similar
- **Image Scale** - Simple resize option
- (Optional) **KSampler** for hi-res fix

Experiments to Try:

1. Generate at 512x512, then upscale to 2048x2048
2. Compare different upscale models (4x-UltraSharp, ESRGAN, etc.)
3. Try upscaling with vs without a refinement pass
4. Upscale 2x then 2x again vs 4x once - any difference?
5. Adjust denoise on the refinement KSampler (0.3 to 0.7)

What to Notice:

- Does upscaling add details or just make it bigger?
 - Which upscaler preserves faces best?
 - How much does VRAM usage increase?
-

Workflow 5: Multi-Model Pipeline

Goal: Use different models for different stages

New Concepts:

- Loading multiple checkpoints
- Switching models mid-generation
- Using specialized models (like inpainting or refiner models)

Experiments to Try:

1. Generate base image with one model, refine with another
2. Use SDXL base + SDXL refiner in sequence
3. Try different model combinations - which pair well?
4. Load 3+ models and see your VRAM usage (remember: you have 32GB!)
5. Use a style-specific model for initial gen, generic model for upscale

What to Notice:

- Do certain models work better for base vs refinement?
 - How much slower is multi-model vs single model?
 - Can you see where one model "hands off" to another?
-

General Experimentation Tips

The Scientific Method:

1. **Change ONE thing** - Don't modify multiple settings at once
2. **Document results** - Screenshot or save with descriptive filenames
3. **Compare side-by-side** - Use the same seed to isolate variables
4. **Ask "why"** - If something looks weird, it's teaching you boundaries

When Things Go Wrong:

- **Red nodes** = Missing models/inputs - check error messages
- **Black images** = VAE issue, try a different VAE or "baked in VAE" model
- **Out of memory** = Model too large, try closing and restarting ComfyUI
- **Weird artifacts** = Try different sampler, adjust CFG, or reduce steps

Node Connection Rules:

- **Orange** (latent) connects to orange
 - **Purple** (image) connects to purple
 - **White** (text/clip) connects to white
 - Can't connect wrong types - ComfyUI prevents this!
-

Resources for Finding Workflows

1. **Civitai** - Click any model → "Creator's Resources" or image → "Workflow"
 2. **OpenArt.ai/workflows** - Searchable ComfyUI workflow library
 3. **ComfyUI Examples** - In your installation folder
 4. **Reddit r/comfyui** - Weekly workflow sharing threads
 5. **ComfyUI Discord** - #workflow-sharing channel
-

Your Learning Journal Template

Date: _____
Workflow: _____
What I Changed: _____
What Happened: _____
Unexpected Result: _____
Next Thing to Try: _____

Progress Milestones

- ☐ Successfully loaded and ran Workflow 1
- ☐ Changed 5 different settings and saw results
- ☐ Made something unexpectedly cool
- ☐ Troubleshooted a red node error
- ☐ Combined two techniques from different workflows
- ☐ Created a workflow from scratch (even a simple one!)
- ☐ Helped someone else understand a node

Remember: Every "broken" workflow teaches you what NOT to do. Every weird result shows you the boundaries. Every successful experiment builds your intuition. Have fun breaking things!