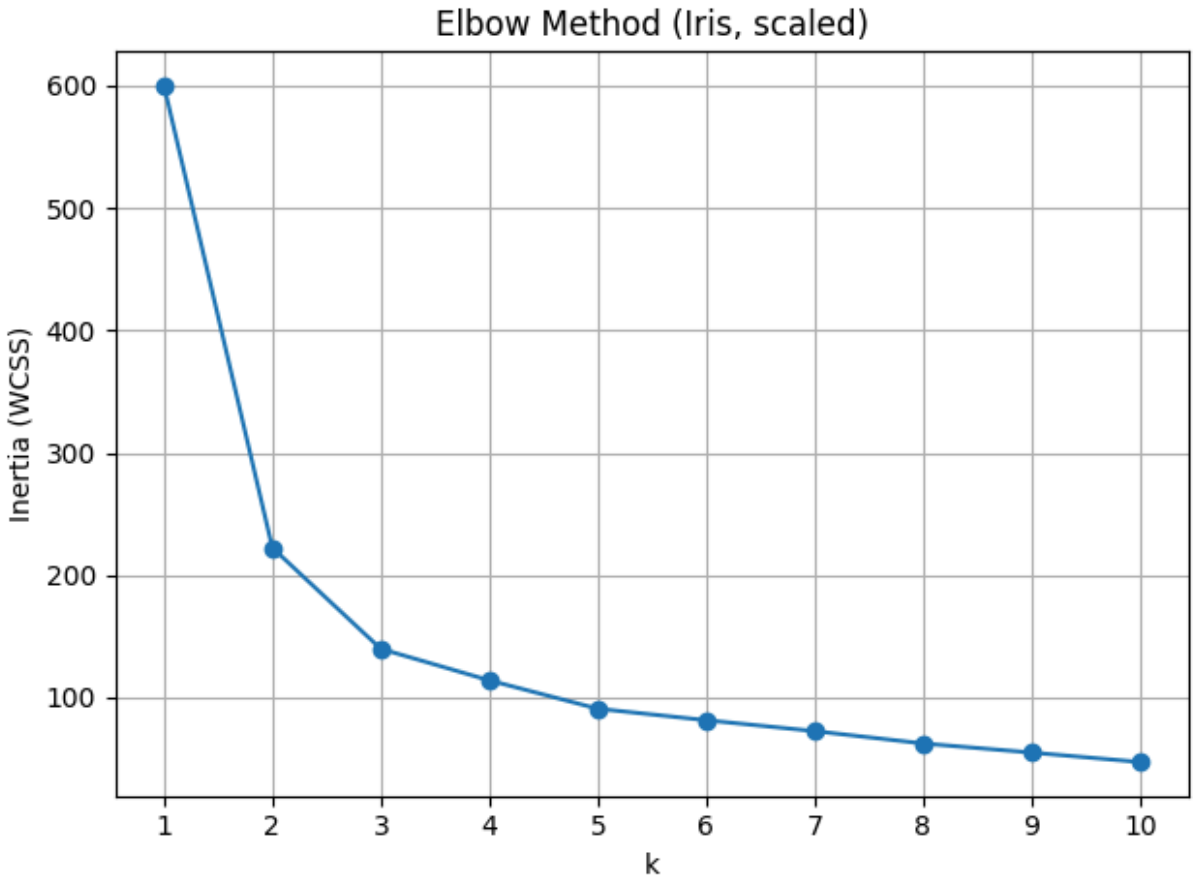
In [73]:

```
"""
K-Means on the Iris dataset
- Uses scaling, elbow, and a 2D scatter on two features
"""

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# ------------------------------
# 1) Load data
# ------------------------------
iris = load_iris()
X = iris.data                          # shape (150, 4)
feature_names = iris.feature_names     # 4 feature names

df = pd.DataFrame(X, columns=feature_names)

print("Samples:", X.shape[0], "| Features:", X.shape[1])
print("Feature names:", feature_names)

# ------------------------------
# 2) Scale features
# ------------------------------
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# ------------------------------
# 3) Elbow method (find k)
# ------------------------------
ks = range(1, 11)
inertias = []
for k in ks:
    km = KMeans(n_clusters=k, init="k-means++", n_init=10, random_state=42)
    km.fit(X_scaled)
    inertias.append(km.inertia_)

plt.figure()
plt.plot(ks, inertias, marker="o")
plt.title("Elbow Method (Iris, scaled)")
plt.xlabel("k")
plt.ylabel("Inertia (WCSS)")
plt.xticks(list(ks))
plt.grid()
plt.tight_layout()
plt.show()

# Choose k (for Iris, k=3 is natural)
k = 3

# ------------------------------
```

```python
# 4) Fit final K-Means (k=3)
# ------------------------------
kmeans = KMeans(n_clusters=k, init="k-means++", n_init = 10, random_state=10
labels = kmeans.fit_predict(X_scaled)

print("\nCluster sizes:", np.bincount(labels))
print("Final inertia (WCSS):", kmeans.inertia_)
print("Centroids in scaled space:\n", kmeans.cluster_centers_)

# ------------------------------
# 5) Simple 2D scatter (pick any two features)
#    just plot two original features
# ------------------------------
# Choose two feature indices to plot
i, j = 0, 1  # sepal length vs sepal width often shows structure

plt.figure()
for c in range(k):
    mask = (labels == c)
    plt.scatter(X[mask, i], X[mask, j], label=f"cluster {c}", s=40)
plt.xlabel(feature_names[i])
plt.ylabel(feature_names[j])
plt.title("K-Means clusters on two original features")
plt.grid()
plt.legend()
plt.tight_layout()
plt.show()

# ------------------------------
# 6) Predict new samples
# ------------------------------
new_samples = np.array([
    [5.0, 3.5, 1.3, 0.3],  # Setosa-like
    [6.0, 2.7, 5.1, 1.6],  # Versicolor/Virginica-like
])
new_scaled = scaler.transform(new_samples)
pred = kmeans.predict(new_scaled)
print("\nPredicted clusters for new samples:", pred)
```

```
Samples: 150 | Features: 4
Feature names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (c
m)', 'petal width (cm)']
```




```
Cluster sizes: [53 50 47]
Final inertia (WCSS): 139.82049635974982
Centroids in scaled space:
 [[-0.05021989 -0.88337647  0.34773781  0.2815273 ]
 [-1.01457897  0.85326268 -1.30498732 -1.25489349]
 [ 1.13597027  0.08842168  0.99615451  1.01752612]]
```


```
Predicted clusters for new samples: [1 0]
```

In [ ]: