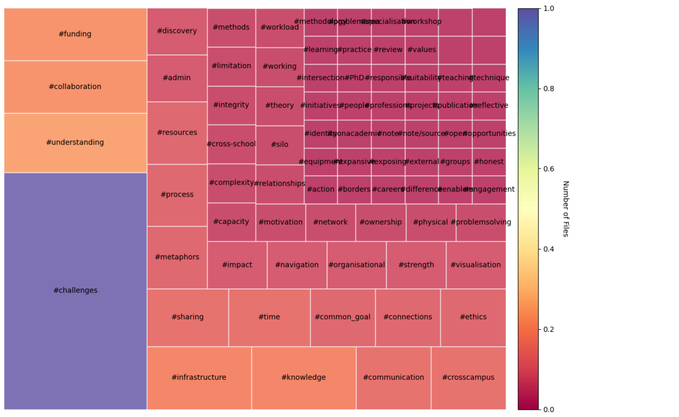
In [64]:
```
import matplotlib.pyplot as plt
import squarify
import matplotlib
import matplotlib.cm as cm
import numpy as np
from wordcloud import WordCloud
```

In [65]:
```
# List of tags and their respective counts
tags = ["#challenges", "#understanding", "#collaboration", "#funding", "#infrastructure", "#knowledge",
        "#communication", "#crosscampus", "#sharing", "#time", "#common_goal", "#connections", "#ethics",
        "#metaphors", "#process", "#resources", "#admin", "#discovery", "#impact", "#navigation", "#organisational",
        "#strength", "#visualisation", "#capacity", "#complexity", "#cross-school", "#integrity", "#limitation",
        "#methods", "#motivation", "#network", "#ownership", "#physical", "#problemsolving", "#relationships",
        "#silo", "#theory", "#working", "#workload", "#action", "#borders", "#careers", "#difference", "#enablers",
        "#engagement", "#equipment", "#expansive", "#exposing", "#external", "#groups", "#honest", "#identity",
        "#initiatives", "#intersection", "#leaning", "#methodology", "#nonacademic", "#note", "#note/source", "#open",
        "#opportunities", "#people", "#PhD", "#practice", "#problemarea", "#professions", "#projects", "#publication",
        "#reflective", "#responsible", "#review", "#specialisation", "#suitability", "#teaching", "#technique",
        "#values", "#workshop"]
```

In [66]:
```
numfiles = [36, 9, 8, 8, 7, 7, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

In [67]:
```
# Normalize the sizes to add up to 1
norm_numfiles = [i/sum(numfiles) for i in numfiles]
```

In [68]:
```
# Ensure plots are displayed inline
%matplotlib inline

# Adjust the overall font size
plt.rcParams.update({'font.size': 10})  # Adjust this value as necessary

# Normalize the numfiles for color mapping
norm_numfiles = np.array(numfiles) / max(numfiles)

# Create a color palette, mapped to these values
cmap = cm.get_cmap('Spectral')  # use 'Spectral' colormap for more color variation
colors = cmap(norm_numfiles)

# Create a treemap
plt.figure(figsize=(15, 10))  # makes the figure larger
squarify.plot(sizes=norm_numfiles, label=tags, alpha=.8, color=colors, edgecolor='white')

# Add a color bar to the plot
cbar = plt.colorbar(cm.ScalarMappable(cmap=cmap), orientation='vertical', pad=0.02)
cbar.set_label('Number of Files', rotation=270, labelpad=20)

# Remove the axes
plt.axis('off')

# Display the plot
plt.show()
```

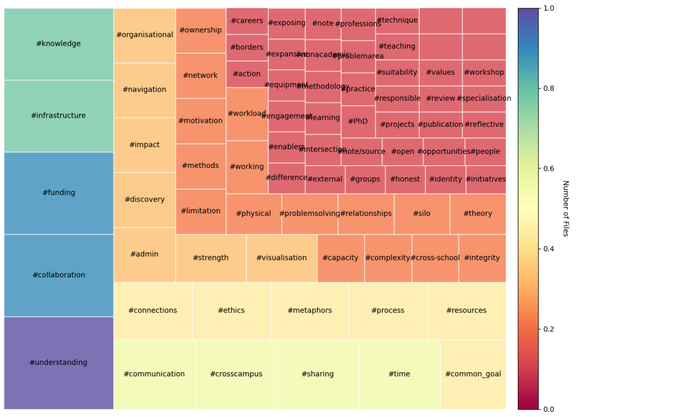


In [69]:
```
# Remove the biggest catagory and replot to see the other data more clearly
# Find the index of the maximum value in numfiles
max_index = numfiles.index(max(numfiles))
```

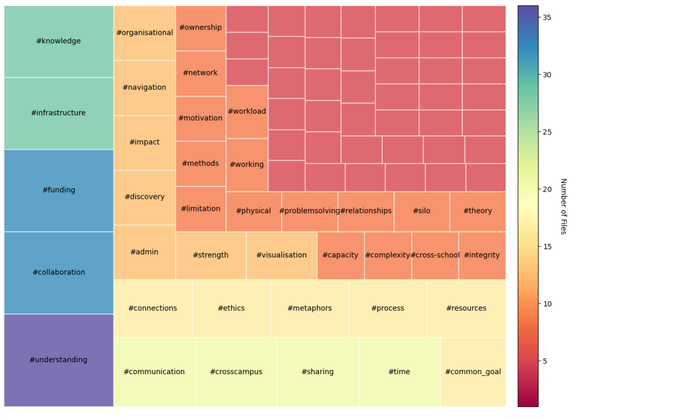
In [70]:
```
# Remove the corresponding elements from tags and numfiles
tags.pop(max_index)
numfiles.pop(max_index)
```

Out[70]:
```
36
```

In [71]:
```
# Normalize the numfiles for color mapping
norm_numfiles = np.array(numfiles) / max(numfiles)

# Create a color palette, mapped to these values
cmap = cm.get_cmap('Spectral')  # use 'Spectral' colormap for more color variation
colors = cmap(norm_numfiles)

# Create a treemap
plt.figure(figsize=(15, 10))  # makes the figure larger
squarify.plot(sizes=norm_numfiles, label=tags, alpha=.8, color=colors, edgecolor='white')

# Add a color bar to the plot
cbar = plt.colorbar(cm.ScalarMappable(cmap=cmap), orientation='vertical', pad=0.02)
cbar.set_label('Number of Files', rotation=270, labelpad=20)

# Remove the axes
plt.axis('off')

# Display the plot
plt.show()
```



In [72]:
```
# Create a new labels list with labels for categories with numfiles > 1
labels = [tag if num > 1 else '' for tag, num in zip(tags, numfiles)]

# Create a treemap with the new labels
plt.figure(figsize=(15, 10))  # makes the figure larger
squarify.plot(sizes=norm_numfiles, label=labels, alpha=.8, color=colors, edgecolor='white')

# Add a color bar to the plot
cbar = plt.colorbar(cm.ScalarMappable(cmap=cmap, norm=plt.Normalize(min_numfiles, max_numfiles)), orientation='vertical', pad=0.02)
cbar.set_label('Number of Files', rotation=270, labelpad=20)

# Remove the axes
plt.axis('off')

# Display the plot
plt.show()
```

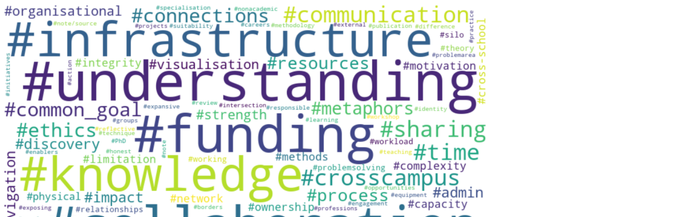


In [73]:
```
# Create a dictionary where the keys are the tags and the values are the numfiles
tag_dict = {tag: num for tag, num in zip(tags, numfiles)}
```

In [74]:
```
# Create a word cloud object and generate a word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white')
wordcloud.generate_from_frequencies(tag_dict)
```

Out[74]:
```
<wordcloud.wordcloud.WordCloud at 0x7f79103d0790>
```

In [76]:
```
# Display the generated word cloud
plt.figure(figsize=(12, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



In [ ]:
```
```