In [12]:
```
import pandas as pd
from astroquery.mast import Tesscut
from astropy.coordinates import SkyCoord
import astropy.units as u
import os
import numpy as np
import matplotlib.pyplot as plt
import lightkurve as lk
from astropy.io import fits
```

Gathering data

Get list of objects

Stars without exoplanets

Using the exoplanet archive, I downloaded filtered the Mission Stars + exocat table to show stars with 0 exoplanets. Downloaded csv as: 'mission_exocat_2023.08.06_09.22.13.csv'

https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=mission_exocat

Stars with exoplanets

This table contains a list of confirmed exoplanets, their star host and their coordinates.

https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=PSCompPars

List of coordinates (RA and Dec)

In order to utlize the atro.query tools to download light curve cutouts, I needed a list of coordinates to feed. Below is the code I used to separate stars with/without exoplanets, then strip the rows down to the correctly formated coordinates so they can be exported to csv.

In [2]:
```
comment_char = '#'
without_exo = pd.read_csv('mission_exocat_2023.08.06_09.22.13.csv', comment=comment_char)
```

In [3]:
```
without_exo
```

Out[3]:

	star_name	st_ppnum	rastr	ra	decstr	dec
0	HIP 57	0	00h00m40.39s	0.168286	-69d40m32.9s	-69.675801
1	HIP 169 A	0	00h02m08.41s	0.535021	-68d16m48.7s	-68.280206
2	HIP 171	0	00h02m09.65s	0.540188	+27d05m04.2s	27.084489
3	HIP 263	0	00h03m19.02s	0.829238	+04d41m13.7s	4.687136
4	HIP 375	0	00h04m40.15s	1.167294	+34d16m17.4s	34.271506
...	...	...	...	...	...	...
2391	HIP 118281 A	0	23h59m29.33s	359.872224	+33d43m26.9s	33.724133
2392	HIP 118310	0	23h59m47.82s	359.949257	+06d39m52.4s	6.664565
2393	GJ 338 B	1	09h14m26.19s	138.609130	+52d41m16.7s	52.687971
2394	HIP 120148	0	20h03m00.82s	300.753400	+20d05m49.8s	20.097161
2395	GJ 317	2	08h40m59.22s	130.246730	-23d27m22.6s	-23.456289

2396 rows × 6 columns

In [4]:
```
without_exo.drop(columns=['star_name','st_ppnum','ra', 'dec'], inplace=True)
without_exo.rename(columns={'rastr': 'RA', 'decstr':'Dec'}, inplace=True)
without_exo
```

Out[4]:

	RA	Dec
0	00h00m40.39s	-69d40m32.9s
1	00h02m08.41s	-68d16m48.7s
2	00h02m09.65s	+27d05m04.2s
3	00h03m19.02s	+04d41m13.7s
4	00h04m40.15s	+34d16m17.4s
...	...	...
2391	23h59m29.33s	+33d43m26.9s
2392	23h59m47.82s	+06d39m52.4s
2393	09h14m26.19s	+52d41m16.7s
2394	20h03m00.82s	+20d05m49.8s
2395	08h40m59.22s	-23d27m22.6s

2396 rows × 2 columns

In [6]:
```
comment_char = '#'
with_exo = pd.read_csv('PSCompPars_2023.08.12_13.06.23.csv', comment=comment_char)
with_exo
```

Out[6]:

	rastr	ra	decstr	dec
0	12h20m42.91s	185.178779	+17d47m35.71s	17.793252
1	15h17m05.90s	229.274595	+71d49m26.19s	71.823943
2	23h31m17.80s	352.824150	+39d14m09.01s	39.235837
3	16h10m24.50s	242.602101	+43d48m58.90s	43.816362
4	19h41m51.75s	295.465642	+50d31m00.57s	50.516824
...	...	...	...	...
5491	01h36m47.60s	24.198353	+41d24m13.73s	41.403815
5492	01h36m47.60s	24.198353	+41d24m13.73s	41.403815
5493	01h36m47.60s	24.198353	+41d24m13.73s	41.403815
5494	11h36m56.93s	174.237219	-00d49m24.83s	-0.823564
5495	19h54m14.99s	298.562449	+08d27m39.98s	8.461105

5496 rows × 4 columns

In [7]:
```
with_exo.drop_duplicates(inplace=True)
with_exo.drop(columns=['ra', 'dec'], inplace=True)
with_exo.rename(columns={'rastr': 'RA', 'decstr':'Dec'}, inplace=True)
with_exo
```

Out[7]:

	RA	Dec
0	12h20m42.91s	+17d47m35.71s
1	15h17m05.90s	+71d49m26.19s
2	23h31m17.80s	+39d14m09.01s
3	16h10m24.50s	+43d48m58.90s
4	19h41m51.75s	+50d31m00.57s
...	...	...
5486	01h44m02.23s	-15d56m01.68s
5490	07h11m08.33s	+30d14m41.84s
5491	01h36m47.60s	+41d24m13.73s
5494	11h36m56.93s	-00d49m24.83s
5495	19h54m14.99s	+08d27m39.98s

4096 rows × 2 columns

Query Light Curves, return .Fits files

Astro Query provides methods for extracting light curves. The coordinates from the previous csv's need to be reformated for the Tesscut.get_cutout parameters.

Documentation: https://astroquery.readthedocs.io/en/latest/api/astroquery.mast.TesscutClass.html#astroquery.mast.TesscutClass.get_cutouts

In [ ]:
```
# Load the CSV
df = pd.read_csv('withexo/withExo.csv') #this was run for each csv

# Convert RA, Dec from hms, dms format to decimal degrees
def convert_to_deg(ra_str, dec_str):
    coord = SkyCoord(ra_str, dec_str, frame='icrs')
    return coord.ra.degree, coord.dec.degree

for index, row in df.iterrows():
    ra_deg, dec_deg = convert_to_deg(row['RA'], row['Dec'])
    coord = SkyCoord(ra_deg * u.deg, dec_deg * u.deg, frame='icrs')

    # Fetch the light curve data as a FITS file
    try:
        hdu_list = Tesscut.get_cutouts(coordinates=coord, size=5)  # adjust the size as needed

        for idx, hdu in enumerate(hdu_list):
            hdu.writeto(f'light_curve_{index}_{idx}.fits', overwrite=True)

    except Exception as e:
        print(f"Could not download data for RA={ra_deg}, Dec={dec_deg}. Reason: {e}")
```

Fold light curves, write jpeg files

In [ ]:
```
def create_folded_lc_plot(filename, output_directory, output_filename):
    # Open the FITS file
    with fits.open(filename) as hdulist:
        # Extract data from the FITS file
        data = hdulist[1].data

        # Extract time and flux columns
        time = data['TIME']
        flux_sum = np.sum(data['FLUX'], axis=(1, 2))

    # Create folded light curve
    lc = lk.FoldedLightCurve(data=data, time=time, flux=flux_sum, flux_err=data['FLUX_ERR'])

    # Create the plot
    fig, ax = plt.subplots(figsize=(2, 2))
    lc.plot(ax=ax, marker='.', linestyle='none', color='blue', xlabel='Phase', ylabel='Flux', title='Folded Light Curve')
    ax.tick_params(axis='both', which='both', length=0)
    ax.set_xticklabels([])
    ax.set_yticklabels([])

    # Save the plot as a JPEG image
    output_path = os.path.join(output_directory, output_filename)
    plt.savefig(output_path, dpi=128)
    plt.close()  # Close the figure to free up memory

# Define the directories and filenames
input_directories = ['star_neg']
output_directories = ['data_main/exo_neg']

# Iterate over the input directories
for input_dir, output_dir in zip(input_directories, output_directories):
    files = os.listdir(input_dir)
    for i, filename in enumerate(files):
        if filename == '.DS_Store':
            continue
        # Set the output filename with leading zeros (e.g., 001.jpg)
        output_filename = f"{i+1:03}.jpg"
        print(filename)
        # Create the folded light curve plot
        create_folded_lc_plot(os.path.join(input_dir, filename), output_dir, output_filename)

    print(f"Processed {len(files)} files in {input_dir} and saved to {output_dir}")
```

In [ ]:
```

```