

# 計算機実験 2 ～ OpenGLの使い方 ～ 7月4日

高田 雅美@E356  
takata@ics.nara-wu.ac.jp

```

void display(void){
    glClearColor( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.0 );
    glEnable( GL_DEPTH_TEST );
    glColor3f( 1.0, 1.0, 1.0);
}

void myInit( char *progame ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH );
    glutCreateWindow( progame );
}

```

プリミティブ関数

陰面処理

```

void display(void){
    glClearColor( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.0 );
    glEnable( GL_DEPTH_TEST );
    glColor3f( 1.0, 1.0, 1.0);
}

void myInit( char *progame ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH );
    glutCreateWindow( progame );
}

```

## > GLUT\_DEPTH

### ◆ Zバッファの利用

- ◆ 奥行き用
- ◆ 記憶領域の確保
- ◆ 未宣言の場合

2次元平面分のみ

```

void display(void){
    glClearColor( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.0 );
    glEnable( GL_DEPTH_TEST );
    glColor3f( 1.0, 1.0, 1.0);
}

void myInit( char *progame ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH );
    glutCreateWindow( progame );
}

```

## > GL\_DEPTH\_BUFFER\_BIT

### ◆ Zバッファの初期化

- ◆ 各領域の値を最大化

```

void display(void){
    glClearColor( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.0 );
    glEnable( GL_DEPTH_TEST );
    glColor3f( 1.0, 1.0, 1.0);
}

void myInit( char *progame ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH );
    glutCreateWindow( progame );
}

```

## 陰面処理

### > 有効範囲の指定

- > 奥行きに応じた描画が可能

## 課題5

### > 土星の作成

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glRotatef( theta, 0.0, 1.0, 0.0 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

陰面処理

アニメーション

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

アニメーション用変数

- ▶ 大域変数
- 引数としても定義可能
- ▶ 描画図形を回転
- 角度を変更

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glRotatef( theta, 0.0, 1.0, 0.0 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

アイドルの定義

- ▶ 何も入力がない場合
- キー入力等がない
- ▶ 引数を実行

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glRotatef( theta, 0.0, 1.0, 0.0 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

ダブルバッファの利用

- ▶ 描画のちらつき防止
- 表示用バッファ
- 更新用バッファ

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glRotatef( theta, 0.0, 1.0, 0.0 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

アイドル中の動き

- ▶ 角度thetaを変更
- 0度~360度
- ▶ glutPostRedisplay()
- 描画を作成
- 更新用バッファに保存

```
#include <math.h>
float theta = 0.0;
void display(void){
    glutSolidSphere( 1.0, 20, 20 );
    glColor3f( 0.0, 1.0, 0.0 );
    glRectf( -1.0, -1.5, 1.0, 1.5 );
    glPopMatrix();
    glutSwapBuffers();
}
void idle( void){
    theta = fmod( theta + 0.5, 360 );
    glutPostRedisplay();
}
void myInit( char *progname ){
    glutInitWindowSize( width, height );
    glutInitDisplayMode( GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE );
    glutCreateWindow( progname );
}
int main( int argc, char **argv ){
    glutDisplayFunc( display );
    glutIdleFunc( idle );
    glutMainLoop();
    return( 0 );
}
```

描画面面の切り替え

- ▶ 更新用バッファの内容を表示用バッファへ

## 課題6

- 中心に太陽
- 太陽系の惑星のアニメーション
  - ◆ 公転は、時計回り
  - ◆ 公転速度の比率は、ネットで検索
    - ◆ 一番遅い星が1周するのに数秒で調整
  - ◆ 課題の確認では、太陽の周りを全ての星が1周するところまで確認