```java
package hyperDap.base.testHelpers;

import static org.junit.Assert.assertEquals;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.MonthDay;
import java.time.OffsetDateTime;
import java.time.OffsetTime;
import java.time.Year;
import java.time.YearMonth;
import java.time.ZoneId;
import java.time.ZoneOffset;
import java.time.ZonedDateTime;
import org.junit.jupiter.api.Test;
import hyperDap.base.helpers.Parser;

public class testParser {

  @Test
  void parseBoolean() {
    Boolean test = true;
    assertEquals(Parser.parse("true", Boolean.class), test);
  }

  @Test
  void parseByte() {
    Byte test = 123;
    assertEquals(Parser.parse("123", Byte.class), test);
  }

  @Test
  void parseShort() {
    Short test = 123;
    assertEquals(Parser.parse("123", Short.class), test);
  }

  @Test
  void parseInt() {
    Integer test = 123;
    assertEquals(Parser.parse("123", Integer.class), test);
  }

  @Test
  void parseLong() {
    Long test = Long.valueOf(123);
    assertEquals(Parser.parse("123", Long.class), test);
  }

  @Test
  void parseBigInt() {
    BigInteger test = BigInteger.valueOf(123);
    assertEquals(Parser.parse("123", BigInteger.class), test);
  }

  @Test
  void parseFloat() {
    Float test = (float) 123.45;
    assertEquals(Parser.parse("123.45", Float.class), test);
  }

  @Test
  void parseDouble() {
    Double test = 123.45;
    assertEquals(Parser.parse("123.45", Double.class), test);
  }

  @Test
  void parseBigDec() {
    BigDecimal test = BigDecimal.valueOf(123.45);
    assertEquals(Parser.parse("123.45", BigDecimal.class), test);
  }

  @Test
  void parseString() {
    String test = "Testing";
    assertEquals(Parser.parse("Testing", String.class), test);
  }

  @Test
  void parseYear() {
    Year test = Year.parse("1996");
    assertEquals(Parser.parse("1996", Year.class), test);
  }

  @Test
  void parseYearMonth() {
    YearMonth test = YearMonth.parse("1996-05");
    assertEquals(Parser.parse("1996-05", YearMonth.class), test);
  }

  @Test
  void parseMonthDay() {
    MonthDay test = MonthDay.parse("--05-07");
    assertEquals(Parser.parse("--05-07", MonthDay.class), test);
  }

  @Test
  void parseDate() {
    LocalDate test = LocalDate.parse("1996-05-07");
    assertEquals(Parser.parse("1996-05-07", LocalDate.class), test);
  }

  @Test
  void parseTime() {
    LocalTime test = LocalTime.parse("22:58:03");
    assertEquals(Parser.parse("22:58:03", LocalTime.class), test);
  }

  @Test
  void parseZoneOffset() {
    ZoneOffset test = ZoneOffset.of("-02:00");
    assertEquals(Parser.parse("-02:00", ZoneOffset.class), test);
  }

  @Test
  void parseTimeOffset() {
    OffsetTime test = OffsetTime.parse("22:58:03-02:00");
    assertEquals(Parser.parse("22:58:03-02:00", OffsetTime.class), test);
  }

  @Test
  void parseDateTime() {
    LocalDateTime test = LocalDateTime.parse("1996-05-07T22:58:03");
    assertEquals(Parser.parse("1996-05-07T22:58:03", LocalDateTime.class), test);
  }

  @Test
  void parseOffsetDateTime() {
    OffsetDateTime test = OffsetDateTime.parse("1996-05-07T22:58:03-02:00");
    assertEquals(Parser.parse("1996-05-07T22:58:03-02:00", OffsetDateTime.class),
    test);
  }

  @Test
  void parseZone() {
    ZoneId test = ZoneId.of("Europe/Berlin");
    assertEquals(Parser.parse("Europe/Berlin", ZoneId.class), test);
  }

  @Test
  void parseZoneDateTime() {
    ZonedDateTime test =
    ZonedDateTime.parse("1996-05-07T22:58:03-02:00[Europe/Berlin]");
    assertEquals(Parser.parse("1996-05-07T22:58:03-02:00[Europe/Berlin]",
    ZonedDateTime.class),
        test);
  }
}
```