ENTORNOS DE DESARROLLO

CFGS 1º Desarrollo de Aplicaciones Web. Actividad de configuración previa Eclipse

< WORKSPACES >

En Eclipse, los "workspaces" (espacios de trabajo) son directorios o carpetas que contienen todos los proyectos, configuraciones y preferencias relacionados con un entorno de desarrollo específico. Crear varios workspaces en Eclipse puede ser útil por varias razones:

Separación y aislamiento de proyectos: Para organizar proyectos relacionados de manera lógica. Por ejemplo, puedes tener un workspace para proyectos de la asignatura de Programación, y otro para proyectos de Entornos de Desarrollo. Esto facilita la gestión y la concentración en proyectos específicos sin que se mezclen los archivos y configuraciones, y evitamos que los cambios o errores en un proyecto afecten a otros proyectos.

Colaboración: Si trabajas en diferentes proyectos con diferentes equipos o en diferentes empresas, cada uno de ellos puede tener su propio workspace. Esto permite mantener separadas las configuraciones y las dependencias de cada proyecto y facilita la colaboración en entornos de desarrollo heterogéneos.

Resolución de problemas: Cuando surgen problemas en Eclipse, a veces puede ser útil crear un nuevo workspace para verificar si el problema persiste en un entorno limpio. Esto puede ayudarte a identificar si el problema está relacionado con la configuración de Eclipse o con un proyecto específico.

Experimentación: Si deseas probar nuevas configuraciones, complementos o experimentar con diferentes configuraciones de Eclipse, puedes hacerlo en un workspace separado sin afectar tu configuración principal.

Mantenimiento y limpieza: Con el tiempo, tu workspace principal puede llenarse de proyectos y configuraciones obsoletos. Crear workspaces adicionales te brinda la oportunidad de mantener limpio y ordenado tu entorno de desarrollo principal.

1 - Abrir Eclipse. A través de la ventana que nos aparece, crear una nueva carpeta llamada **“entornos”** en la ruta donde se crean por defecto los Workspaces. Si no aparece dicha ventana, pulsamos en “*File->Switch Workspace-> Other…”*. Crea dentro de esta carpeta "entornos", otra carpeta que se llame **“entornos1” y** seleccionala para que el programa identifique que tiene que crear un workspace en dicha carpeta. Haz la misma operación pero creando el workspace **“entornos2”.**

2 - Prueba a cambiar de “Workspace”, “*File->Switch Workspace”*, entre los que ya tuvieras y el nuevo que has creado.

< JAVA DEVELOPMENT KIT >

Es un conjunto de herramientas y software proporcionado por Oracle (anteriormente por Sun Microsystems) para desarrolladores que trabajan con Java. Es esencial para crear, compilar, depurar y ejecutar aplicaciones Java. Aquí tienes algunas de las principales funciones y componentes del JDK:

Compilación de código fuente: El JDK incluye el compilador Java (javac), que se utiliza para traducir el código fuente Java escrito por los programadores en código binario comprensible para la máquina virtual Java (JVM).

Bibliotecas estándar: El JDK proporciona un conjunto de bibliotecas estándar de Java (también conocidas como Java API) que contienen clases y métodos predefinidos que los desarrolladores pueden utilizar en sus aplicaciones. Estas bibliotecas abarcan una amplia gama de funcionalidades, como manipulación de archivos, entrada y salida, comunicaciones en red, gráficos, y más.

Herramientas de depuración: El JDK incluye herramientas para identificar y solucionar problemas en sus aplicaciones, como el depurador Java (jdb) y herramientas de monitoreo de rendimiento.

JRE (Java Runtime Environment): El JDK contiene una copia del JRE, que es necesario para ejecutar aplicaciones Java en una máquina. Esto permite a los desarrolladores probar sus aplicaciones antes de distribuirlas.

Documentación y recursos de desarrollo: El JDK incluye documentación detallada sobre las clases y métodos disponibles en las bibliotecas estándar de Java, así como recursos y herramientas para ayudar a crear aplicaciones Java de manera eficiente.

En resumen, el JDK es esencial para cualquier persona que desee desarrollar aplicaciones en Java. Proporciona las herramientas y recursos necesarios para crear, compilar, depurar y ejecutar programas Java de manera efectiva y eficiente.

3 – Configuración JDK: Tenemos que asegurarnos que Eclipse nos ha reconocido correctamente las herramientas JDK. Para ello, vamos a "*Windows->Preferencies->Java->Installed JREs*. Aquí nos aparecerá el que tenemos instalado, y nos dará la opción de instalar nuevos pulsando el botón "add".

< NUEVO PROYECTO, IMPORTAR/EXPORTAR>

4 – Creación de un nuevo proyecto Java: *File -> New -> Other -> Java Proyect.* (desactivar opción "Create module-info.java file". Finish y open perspective.

Vamos a crear 2 proyectos. Por un lado, en el Workspace entornos1, vamos a crear un proyecto que se llame “pruebaentornos1”, y en el Workspace entornos2, vamos a crear un proyecto que se llame “pruebaentornos2.

5 – Vamos a crear un nuevo Workspace, llamado **“entornos3”** Una vez lo tengamos listos, vamos a cambiar al Workspace “entornos3”, y vamos a importar los dos proyectos java que hemos creado anteriormente: “pruebaentornos1” y “pruebaentornos2”: *File -> Import -> General -> Projects from Folder or Archive…* al Workspace **“entornos3”.**

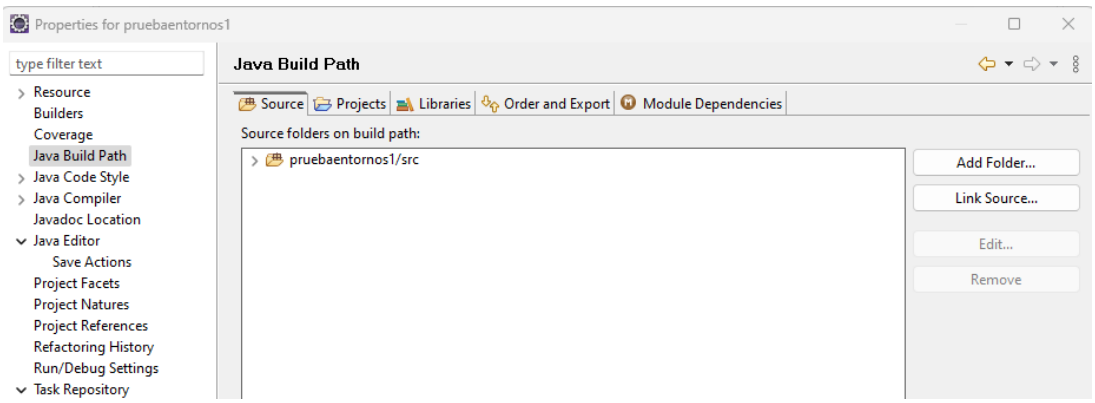
Por tanto, en este nuevo entorno “entornos3” tendremos los 2 proyectos Java que teníamos en los Workspace “entornos1” y “entornos” (Sin embargo, la carpeta nativa de estos proyectos seguirá siendo los “Workspaces” originales, es decir, las carpetas del propio proyecto pruebaentornos1 y pruebaentornos2 seguirán situadas en sus correspondientes “Workspaces” donde se crearon inicialmente, por lo que cualquier cambio futuro se realizará sobre estas carpetas.

Lo siguiente que haremos, es exportar los 2 proyectos del Workspace “entornos3” como un archivo zip, a la carpeta que queramos:
File-> Export -> General -> Archive File -> Seleccionamos los dos proyectos” -> Browse -> Carpeta deseada. Esto nos generará un archivo .zip con los proyectos seleccionados. Si los queremos añadir posteriormente a un workspace, debemos descomprimirlos e importarlos.

< PROPIEDADES DEL PROYECTO – JAVA BUILD PATH >

Tenemos dos formas de llegar a la ventana de propiedades de nuestro proyecto. Por un lado, podemos pulsar con el botón derecho sobre el icono de nuestro proyecto, y seleccionar *“Build Path -> Configure Build Path”*, o también podemos hacer clic sobre la pestaña *“Proyect”, y pulsar en “Properties”*.

Se nos abrirá una nueva ventana, y allí pulsaremos en *“Java Build Path”*



La primera pestaña que nos encontramos es “Source”, donde podemos configurar la ruta para los archivos .java (código fuente), y .class (bytecode que será interpretado pro la máquina virtual de java).

6 – Cambiar el nombre de la carpeta “src” de cada proyecto “pruebaentornos1” y pruebaentornos2”, por los nombres “fuentesprueba1” y “fuentesprueba2”, respectivamente.
Observa a través del explorador de Windows, cómo han cambiado los nombres.
Ahora, puedes probar a cambiar el nombre desde el explorador de Windows, y comprobar si el programa lo reconoce.

La segunda pestaña que nos encontramos es “Libraries”, donde nos aparecerá la librería de Java que estamos utilizando, tal y como hemos visto en apartados anteriores.

La tercera pestaña “order and Export”, determina el orden en que los proyectos y las bibliotecas aparecen en el classpath. Export determina que los proyectos y bibliotecas serán exportados y por lo tanto disponible en otros proyectos que dependen de éste.

< NUEVO PROYECTO – HOLAMUNDO >

Vamos a crear nuestro primer programa. Cambiamos al Workspace Entornos 3, borramos los proyectos actuales pruebaentornos1 y pruebaentornos2, y creamos nuestro primer proyecto Java llamado **“Debug”. Quitamos la opción “Create module-info…”.** A través de la pestaña “Window”, habilitamos la pestaña “Console”. Sobre nuestro proyecto “Debug”, *pulsamos botón derecho-> New -> Class -> Marcamos “public static void…”, y lo llamamos HolaMundo.*
Dentro de public static void main…, añadimos los comandos del programa que crearemos.

```
System.out.println("Hola Mundo");
```

< ACTIVIDAD DEBUG 1 – MAYORDETRES >

Para realizar estas pruebas se creará un proyecto nuevo:

1. Crear una clase que compruebe el mayor de tres números que le pediremos al usuario.
2. Nos dará un mensaje diciendo cuál es el mayor de los 3 números introducidos.

En la clase creada, realizar las siguientes operaciones con ayuda del debug:

- Introducir los números 5, 4 y 3 y "forzar" para que nos diga que el número 3 es el mayor.
- Introducir los números 5, 5 y 5 y "modificar la ejecución" el valor de uno de ellos por 8.
- Incluir un punto condicional para que solo pare en el inicio de las condiciones cuando el número 1 tenga el valor de 6.

- Incluir un punto condicional para que cuando numero1 sea 1, y numero2 sea 2, se detenga. (numero1==1 && numero2==2)

En la misma clase creada, después de la comprobación del número mayor realizamos las siguientes acciones:

1. Comprobar si la resta del primero número menos el segundo introducido es mayor de 1.
2. Si lo es mostrar un mensaje: "La resta de los dos primeros números es mayor que 1".

Con la ayuda del debug:

- Introducir los números 5, 4, 3 y "forzar" para que aparezca el mensaje anterior.

```
import java.util.Scanner;
public class MAYORDETRES {
	public static void main(String[] args) {
		Scanner scanner = new Scanner(System.in);
		System.out.print("Ingrese el primer número: ");
		double numero1 = scanner.nextDouble();
		System.out.print("Ingrese el segundo número: ");
		double numero2 = scanner.nextDouble();
		System.out.print("Ingrese el tercer número: ");
		double numero3 = scanner.nextDouble();
		double mayor = numero1;
		if (numero2 > mayor) {
		   mayor = numero2;
		}
		if (numero3 > mayor) {
		   mayor = numero3;
		}
		System.out.println("El número mayor es: " + mayor);
		if ((numero1-numero3)>1) {
		    System.out.println(numero1 + " - " + numero3 + " > 1");
		    System.out.println("La resta de los dos primeros números es mayor que 1");
		}
		scanner.close();
	  }
}
```

< ACTIVIDAD DEBUG 2 – DIASSEMANA >

1. Crear una clase que nos pregunte por consola un día de la semana y que nos diga si el día es laboral o no laboral. Los días de la semana se introducirán con un número del 1 al 7.

Una vez creada utilizar el debug:

- Sin modificar el código, si introducimos "1" (lunes) nos devuelva "Descanso" y si introducimos "6" (sabado) nos devuelva "Dia laboral!".
- Añadir un punto de ruptura que sólo pare cuando el día introducido sea "Martes"

2. Crear una clase o añadir en la anterior para que nos pida un número por consola y que nos diga si un número es par o impar. Una vez creada, con el debug realizar las siguientes acciones sin modificar el código:

- Añadir un punto de ruptura dentro del if que sólo pare cuando el número que le ha entrado por consola sea 2.
- Añadir un punto de ruptura dentro del else que sólo pare cuando el número que le ha entrado sea 7.
- Añadir al primer punto condicional que pare cuando sea 2 ó 4.

```
import java.util.Scanner;

public class DIASSEMANA {

    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);
        System.out.println("Introduzca un número:");
        int dia = reader.nextInt();

        while (dia > 7) {
            System.out.println("El número introducido no corresponde a ningún día
de la semana");
            System.out.println("Introduzca un número nuevo");
            dia = reader.nextInt();

        }

        if ((dia == 1) || (dia == 2) || (dia == 3) || (dia == 4) || (dia == 5)) {

        System.out.println("EL día seleccionado es laboral");
        }
        else  {
            System.out.println("El día seleccionado es descanso");
        }

        reader.close();
    }
}
```

< ACTIVIDAD DEBUG 3 – SUMAIMPARES >

Crear una clase que genere dos números aleatorios entre 0 y 20 y que nos sume los números impares que hay entre ellos. Además, que muestre en consola los números que ha sumado. Forzar para que introduciendo los números 7 y 3 nos sume los pares hasta 0.

- Incluir un punto de ruptura dentro del bucle para que sólo se pare cuando la suma sea >10
- Incluir un punto de ruptura dentro del bucle for para que sólo pare cuando la suma esté entre 10 y 20.

```
public class SUMAIMPARES {

    public static void main(String[] args) {
        int n1 = (int) (Math.random() * 20);
        int n2 = (int) (Math.random() * 20);
        int suma = 0;

        int min = Math.min(n1, n2);
        int max = Math.max(n1, n2);

        for (int i = min; i <= max; i++) {
            if (i % 2 != 0) {
                suma = suma + i;
            }
        }

        System.out.println("Números Generados: " + n1 + " and " + n2);
        System.out.println("Suma de los números impares entre ellos: " + suma);
    }
}
```

< ACTIVIDAD DEBUG 4 – ADIVINALO >

Construir un programa que genere un número aleatorio entre 1 y 50 y el usuario tenga que acertarlo. Para hacerlo el programa dará 5 oportunidades, si lo acierta antes terminará la ejecución.

Después de cada intento el programa debe indicarle al usuario si el número introducido por él es mayor, menor o igual al número a acertar. También deberá indicar el número de intentos restantes.

Utiliza el debug para dar con la solución correcta.

```
import java.util.Random;
import java.util.Scanner;

public class ADIVINALO {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Random random = new Random();

        int numeroSecreto = random.nextInt(50) + 1; // genera un número entre 1 y
50
        int intentos = 5;

        System.out.println("¡Adivina el número secreto entre 1 y 50!");

        while (intentos > 0) {
            System.out.println("Tienes " + intentos + " intento(s) restante(s).
Introduce tu número:");

            int numeroUsuario;
            try {
                numeroUsuario = scanner.nextInt();

                if (numeroUsuario == numeroSecreto) {
                    System.out.println("¡Felicitaciones! Has adivinado el
número.");
                    return;
                } else if (numeroUsuario < numeroSecreto) {
                    System.out.println("El número secreto es mayor.");
                } else {
                    System.out.println("El número secreto es menor.");
                }

                intentos--;

                if (intentos == 0) {
                    System.out.println("Lo siento, has agotado tus intentos. El
número secreto era: " + numeroSecreto);
                }
            } catch (Exception e) {
                System.out.println("Por favor, introduce un número válido.");
                scanner.next(); // limpia el input no válido
                scanner.close();
            }
        }
    }
}
```

< ACTIVIDAD DEBUG 5 – INTRODUCEUNCERO >

Crear una clase que esté pidiendo número al usuario hasta que se introduzca un 0. Con los números introducidos se realizarán las siguientes acciones:

- Contar los números que son impares
- Contar los números que son pares
- Mostrar el número mayor de todos los introducidos
- Mostrar el número menor de todos los introducidos
- Contar los números divisibles entre 3
- Realizar la media de todos los números introducidos
- Sumar todos los números introducidos
- Listar todos los números que se han introducidos

Utilizar el debug para ir depurando los posibles errores que podamos tener al implementarlo.

Una vez implementado lo anterior, con el debug realizar lo siguiente:

1. Crear dentro del bucle un punto de ruptura que sólo se pare cuando llevemos contados más de 3 números pares.
2. Crear dentro del bucle un punto de ruptura que sólo se pare cuando la suma de los números introducidos supere 20.
3. Añadir un punto de ruptura que pare la ejecución en la 5ª ejecución si las hay.
4. Añadir en expresiones las condiciones de los posibles if que tengas.

```
import java.util.Scanner;

public class INTRODUCEUNCERO {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

	  int numeroU=0;
	  int menor=0;
	  int mayor=0;
	  int suma=0;
	  double media;
	  int contador=-1;
	  int contPares=0;
	  int contImpares=0;

	  Scanner sc = new Scanner(System.in);
	  System.out.println("Introduce un numero");
	  numeroU = sc.nextInt();
	  menor = numeroU;

	  do{

	      //Comprobamos si el numero es menor que el actual
	      if(menor>numeroU){
	          menor = numeroU;
	      }
```

```
            //Comprobamos si el numero es mayor que el actual
            if(mayor<numeroU){
                mayor=numeroU;
            }

            //Suma el numero y lo acumulamos
            suma+=numeroU;

            //Si el numero es positivo, suma a la variable de los positivos
            // y sino a la de los negativos
            if(numeroU%2==0){
                contPares++;
            }else{
                contImpares++;
            }

            //aumento el contador
            contador++;

            //pido un numero al usuario
            System.out.println("Introduce un numero");
            numeroU = sc.nextInt();

            //Cuando el usuario ponga un 0 terminamos

        }while(numeroU!=0);

        //Calculamos la media
        media = (double) suma / contador;
        sc.close();

        System.out.println("Menor es: "+menor);
        System.out.println("Mayor es: "+mayor);
        System.out.println("Suma es: "+suma);
        System.out.println("Pares: "+contPares);
        System.out.println("Impares: "+contImpares);
        System.out.println("Media: "+media);

    }

}
```

< ACTIVIDAD DEBUG 6 – SUMAFACTORES >

En el siguiente código se intenta realizar lo siguiente: dado un número comprobar si la suma de sus factores da como resultado el número. Los factores de un número son los números enteros que pueden multiplicarse para dar como resultado ese número, por ejemplo: los factores de 6 son 1, 2 y 3 --> 1 + 2 +3 = 6. Por lo tanto, el número 6 cumpliría la condición.

Analizar el siguiente código facilitado y comprobar los posibles errores que pueda contener para lograr el resultado deseado. En el siguiente código se comprueban los números que lo cumplen entre 1 y 100. Añadir lo que necesitemos para encontrar los errores, puntos de ruptura, mensajes de salida, visualizar valor de las variables en ejecución, etc....

CÓDIGO CON ERRORES

```
public class SUMAFACTORES {

      public static void main(String[] args) {
      int num = 100;

      for (int i = 1; i <= 10; i++) { //cambio final del for por 10
         int aux = 0;
         for (int n = 1; n < i / 2 + 1; n--) { //cambio sentido del bucle n++ por n--
            if (i % n != 0) { //cambio == por !=
                        aux += n;
            }
         }
         if (aux == i) {
            System.out.println("Encontrado " + i + " ");
         }
      }
   }
}
```

CÓDIGO VÁLIDO

```
public class SUMAFACTORES {

     public static void main(String[] args) {
      int num = 100;

      System.out.println(num +" Números entre 1 y " + num);

      for (int i = 1; i <= num; i++) {
         int aux = 0;
         for (int n = 1; n < i / 2 + 1; n++) {
            if (i % n == 0) {
                        //System.out.println("Añadimos el número:" + n);
                        aux += n;// El divisor divisible se agrega a temp
            }
         }
         if (aux == i) {// Si la suma de los factores es igual al número original, significa que el número está
completo
            System.out.println("Encontrado " + i + " ");// generar el número
         }
      }
   }
}
```

< ACTIVIDAD DEBUG 7 – ADIVINALO2 >

Realizaremos lo contrario al programa “Debug 4 – Adivinalo”. Construir un programa adivine un número que nosotros elijamos. Le tendremos que decir el límite superior en el que se encuentra el número que hemos pensado y después de cada intento el programa nos preguntará si ha acertado, si es menor o es mayor. Si es mayor introduciremos "1", si es menor introduciremos "2", si ha acertado introduciremos "0". El programa tendrá 5 oportunidades, si lo acierta antes terminará la ejecución.

Utiliza el debug para dar con la solución correcta.

```
import java.util.Scanner;
public class ADIVINALO2 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		System.out.println("Introduce el límite superior");
		Scanner sc = new Scanner(System.in);
		int inferior = 1;
	  int limite = sc.nextInt();
		int adivina;

		boolean acertado = false;
		int i = 1;

		while ((i<=5)&&(!acertado)) {

			adivina = (int) (Math.random()* (limite-inferior) + inferior);
			System.out.println(("¿Es " + adivina + "?"));
			System.out.println("Pulsa 1 si es mayor, 2 si es menor y 0 si
es correcto");
			int respuesta = sc.nextInt();

		  if (respuesta == 0) {acertado = true;
		  System.out.println("¡He acertado tu número!. Me debes una caña :-
)");
		  }
		  else if (respuesta == 1) {
			inferior = adivina; //el número pensado es mayor
		  }
		  else if (respuesta == 2){
			limite = adivina; //el número pensado es menor
		  }


		   i++;
		}
		System.out.println("No He sido capaz de acertar tu número, te debo
una caña ;-D");
		sc.close();
	}

}
```

< ACTIVIDAD DEBUG 8 – VALIDARCORREO >

Comprobar y entender el funcionamiento del siguiente programa, e introducir unos puntos de interrupción para que, una vez escaneado el correo electrónico, cambiemos el valor de la variable correspondiente con el fin de que nos indique que el correo electrónico introducido no es correcto.

```
import java.util.Scanner;

public class VALIDARCORREO {

	public static void main(String[] args) {
		String correo;

		Scanner sc = new Scanner(System.in);

		System.out.println("Introduzca el correo a validar:");
		correo = sc.nextLine();

		System.out.println(validarCorreo(correo));
		sc.close();
	}

	public static String validarCorreo(String correo) {
		String mensaje = "";
		boolean valido = true;
		int contArroba=0, contPuntos=0;

		//si no hay ninguna @ no realizamos el proceso
		if ((correo.indexOf('@')==-1) || (correo.indexOf('.')==-1) ||
(correo.indexOf(' ') != -1)){
			valido = false;
		}else {
			for (int i = 0; i < correo.length(); i++) {
				if (correo.charAt(i) == '@') {
					contArroba++;
				}
				if (correo.charAt(i) == '.') {
					contPuntos++;
				}
			}
		}

		if (contArroba > 1 || contPuntos > 1) {
			valido = false;
		}

		if (valido) {
			mensaje = "El correo introducido es válido.";
		}else {
			mensaje = "El correo introducido no es válido.";
		}

		return mensaje;
	}
}
```

< ACTIVIDAD DEBUG 9 – BINARIO >

Para el siguiente programa, deberás solucionar todos los problemas que aparecen como advertencia en la pestaña “Problems”, y todos los errores que se produzcan a la hora de ejecutarlo.

```
import java.util.Iterator;
import java.util.Scanner;

public class decimalabinario {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		int numero;
		String binario;

		Scanner sc = new Scanner(System.in);

		System.out.println("Introduzca el número decimal a convertir");
		numero = sc.nextInt();

		binario = convertirDecimal(numero);

		System.out.println("El número en binario es " + binario);

	}

	public static String convertirDecimal(int numero) {

		String convertido = "";
		int num= numero;
		int resto = 0;

		while (num>2) {
			resto = num%2;
			num = num/2;
			convertido = Integer.toString(resto) + convertido;

		}
		convertido = String.valueOf(num)+convertido;
		return convertido;
	}

}
```