
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	52
--	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	----

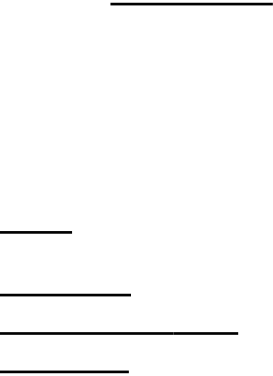
Table of Contents

Code	

ProActive guided tour

This tour is a practical introduction to ProActive.

First you will get some practical experience on how to program using ProActive. This will help your understanding of the library, and of the c 2mpts.



—

Hello world ! example

This example implements a very simple client–server application. A client object display a `String`



Launching the client

```
> java org.objectweb.proactive.examples.hello.HelloClient //localhost/Hello
```

Hello World from abroad: another VM on a different host

Starting the server

Log on to the server's host, and launch the `Hello` class.

```
remoteHost> java org.objectweb.proactive.examples.hello.Hello &
```

Launching the client

Log on to the client Host, and launch the client

```
clientHost> java org.objectweb.proactive.examples.hello.HelloClient //remoteHost/Hello
```

1.2. Initialization of the activity

Active objects, as indicates their name, have an activity of their own (an internal thread).



```
}  
Hello of 28}
```

```
System.out.println("frame is killed");
```

use the (Hello of 28Cor



2.1. Synchronization with ProActive

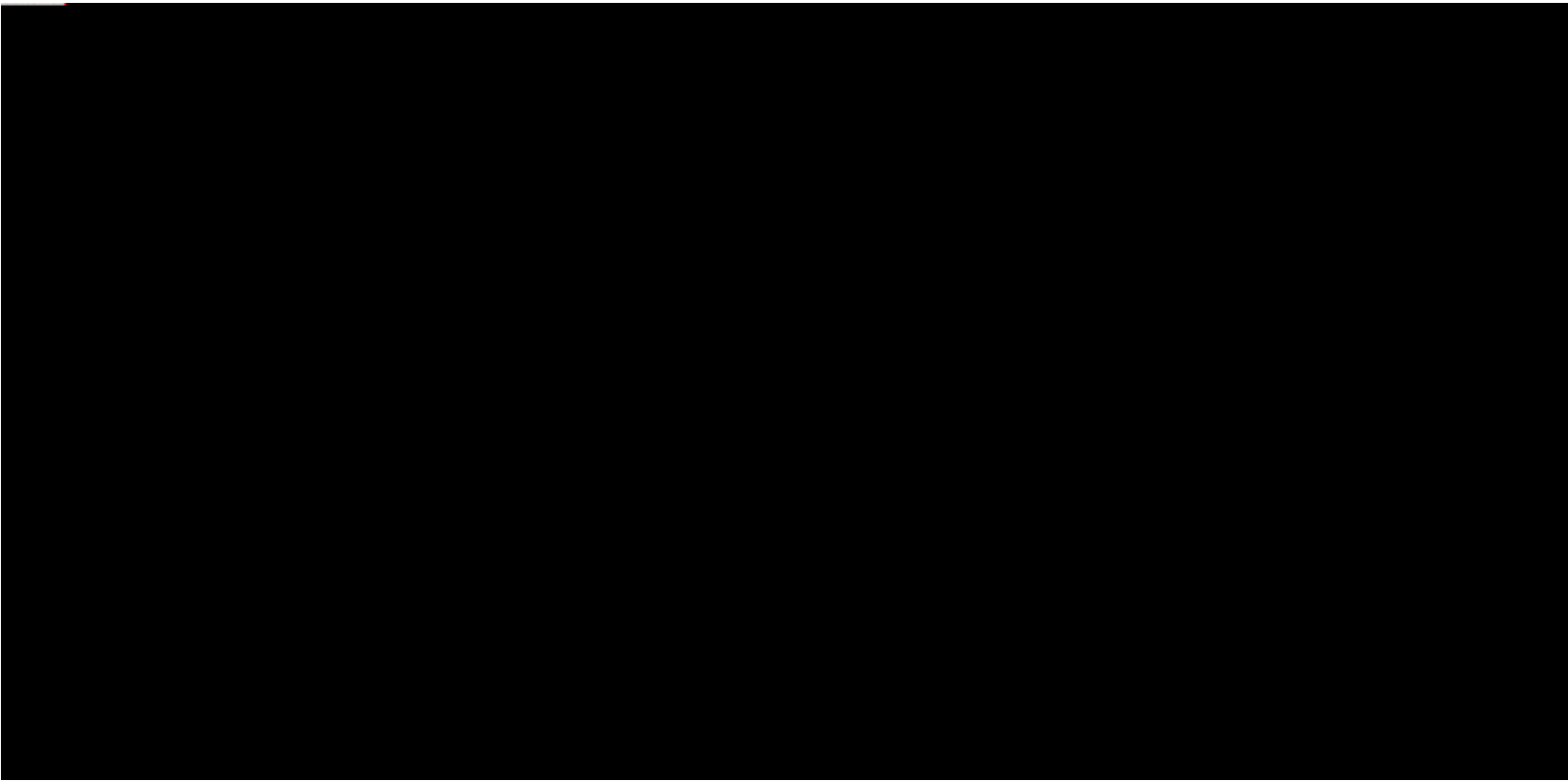
ProActive provides an advanced synchronization mechanism that allows an easy and safe implementation of potentially complex synchronization policies.

This is illustrated by two examples :

- The readers and the writers
- The dining philosophers

The readers–writers

The readers and the writers want to access the same data. In order to allow concurrency while ensuring the consistency of the readings, accesses to the data have to be synchronized upon a specified policy. Thanks to ProActive, the accesses are guaranteed to be allowed sequentially.



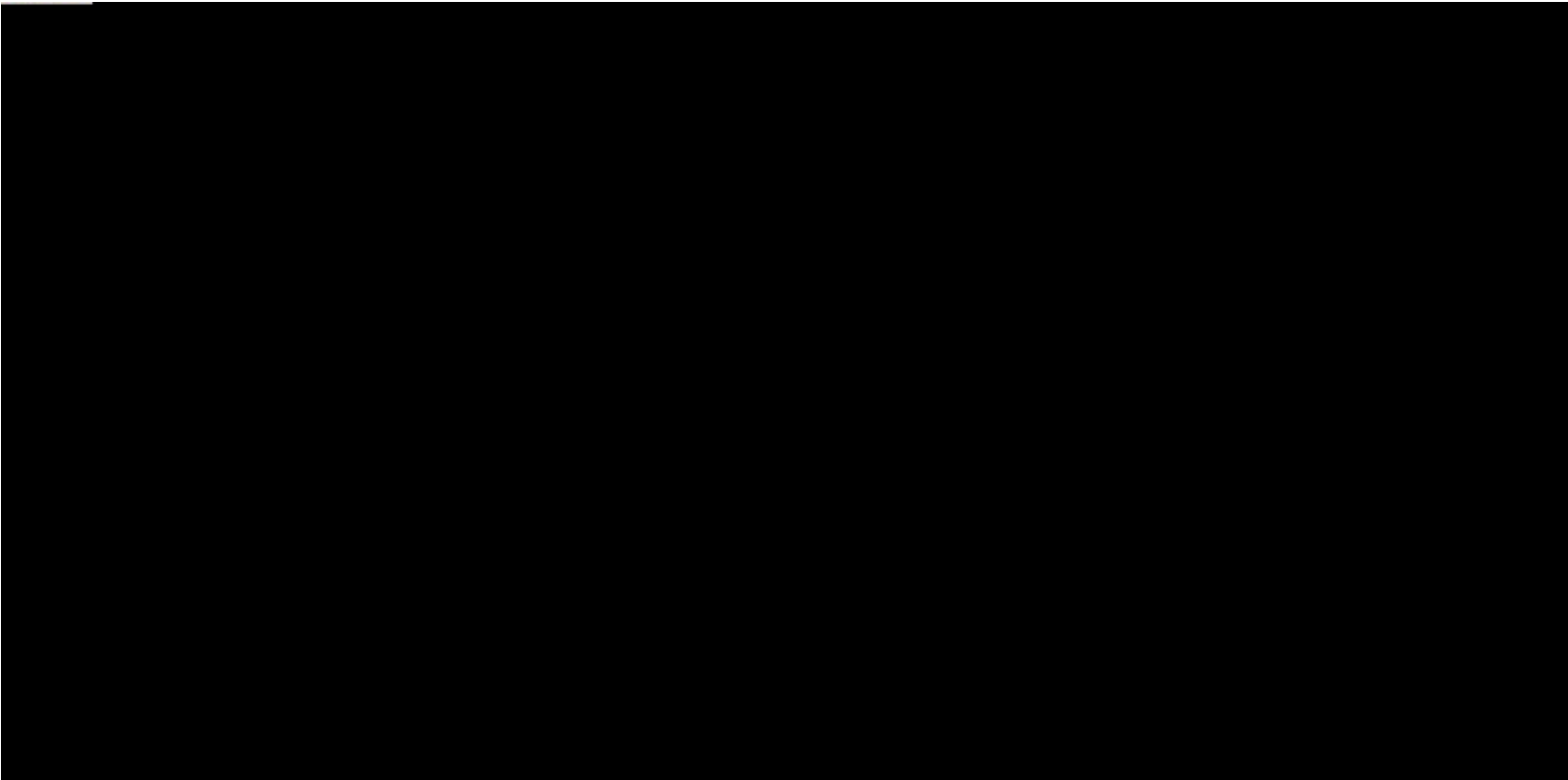
The dining philosophers

The "dining philosophers" problem is a classical exercise in the teaching of concurrent programming. The goal is to avoid deadlocks.

We have provided [an illustration of the solution](#) using ProActive, where all the philosophers are active objects, as well as the table (controller) and the dinner frame (user interface).

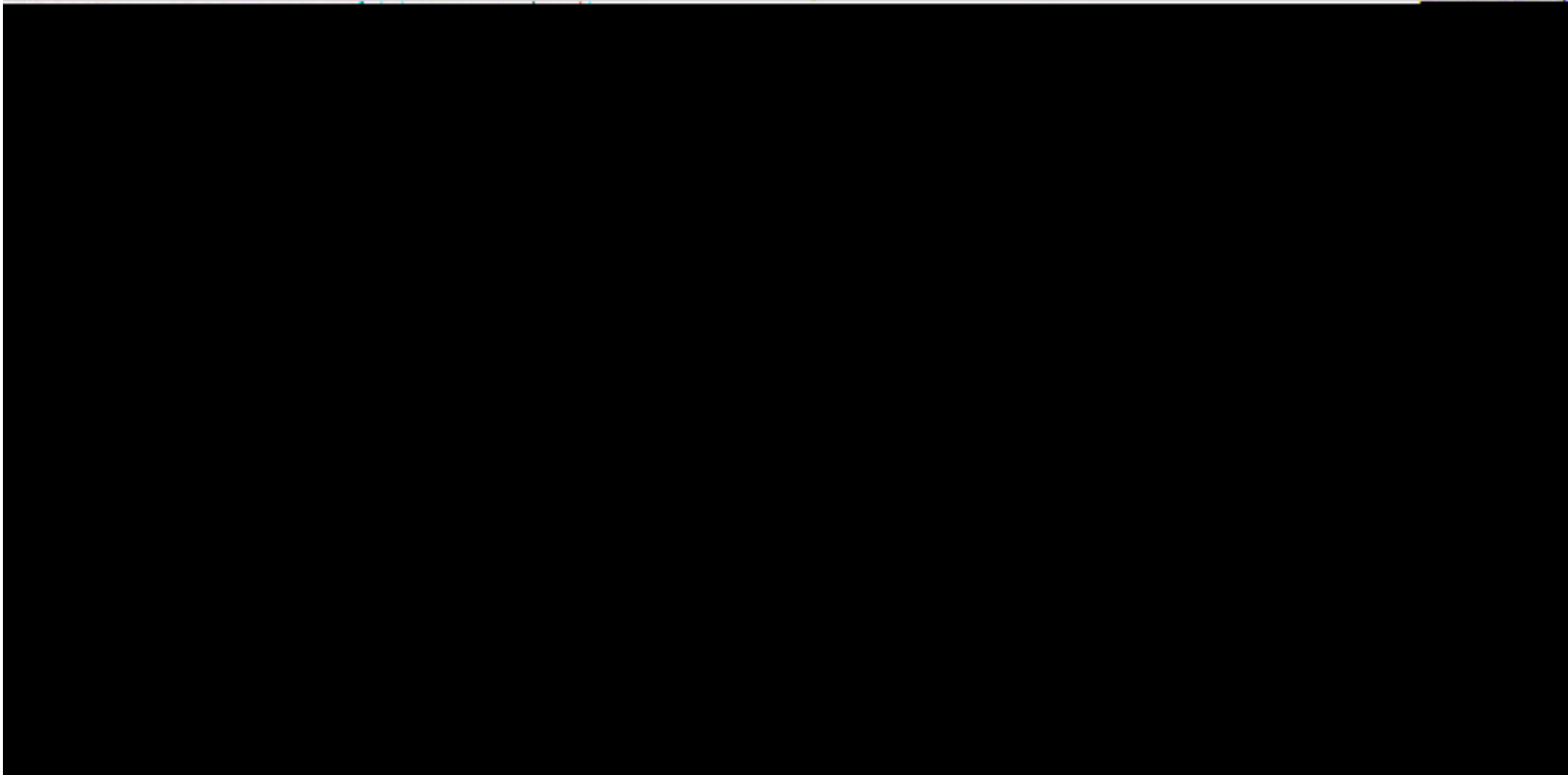
1. start the philosophers application

with philosophers.sh or philosophers.bat



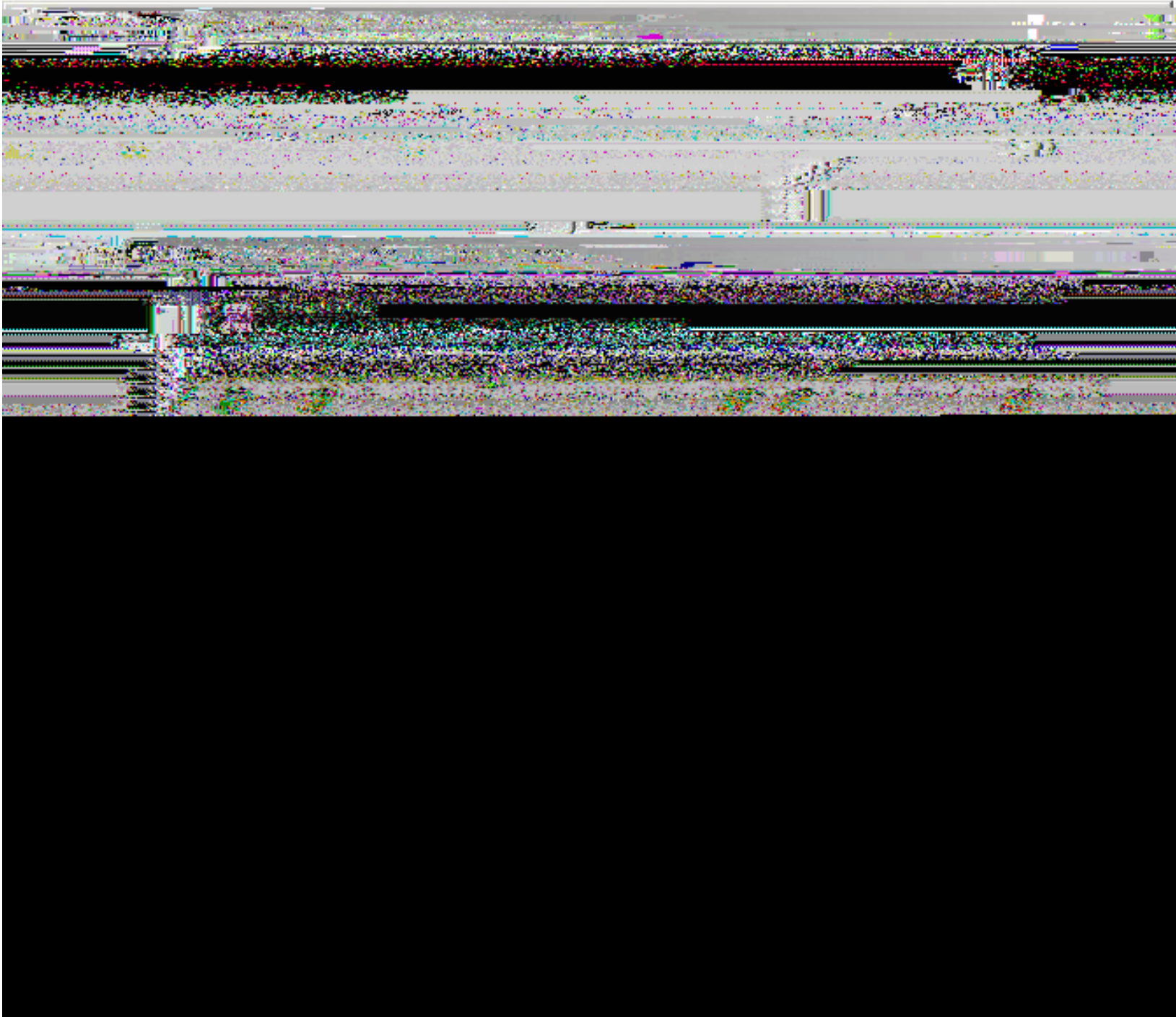
ProActive creates a new node and instantiates the active objects of the application : DinnerLayout, Table, and Philosopher



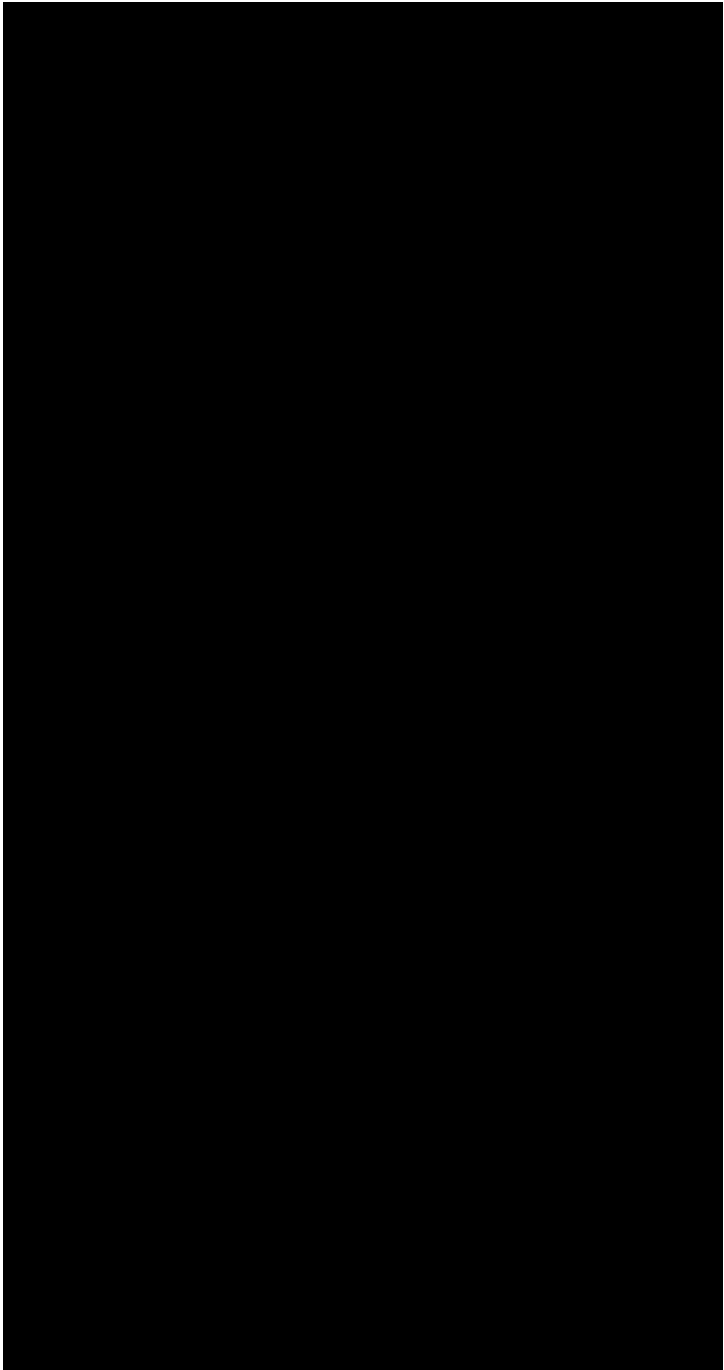


—

—



the 4 renderers are launched

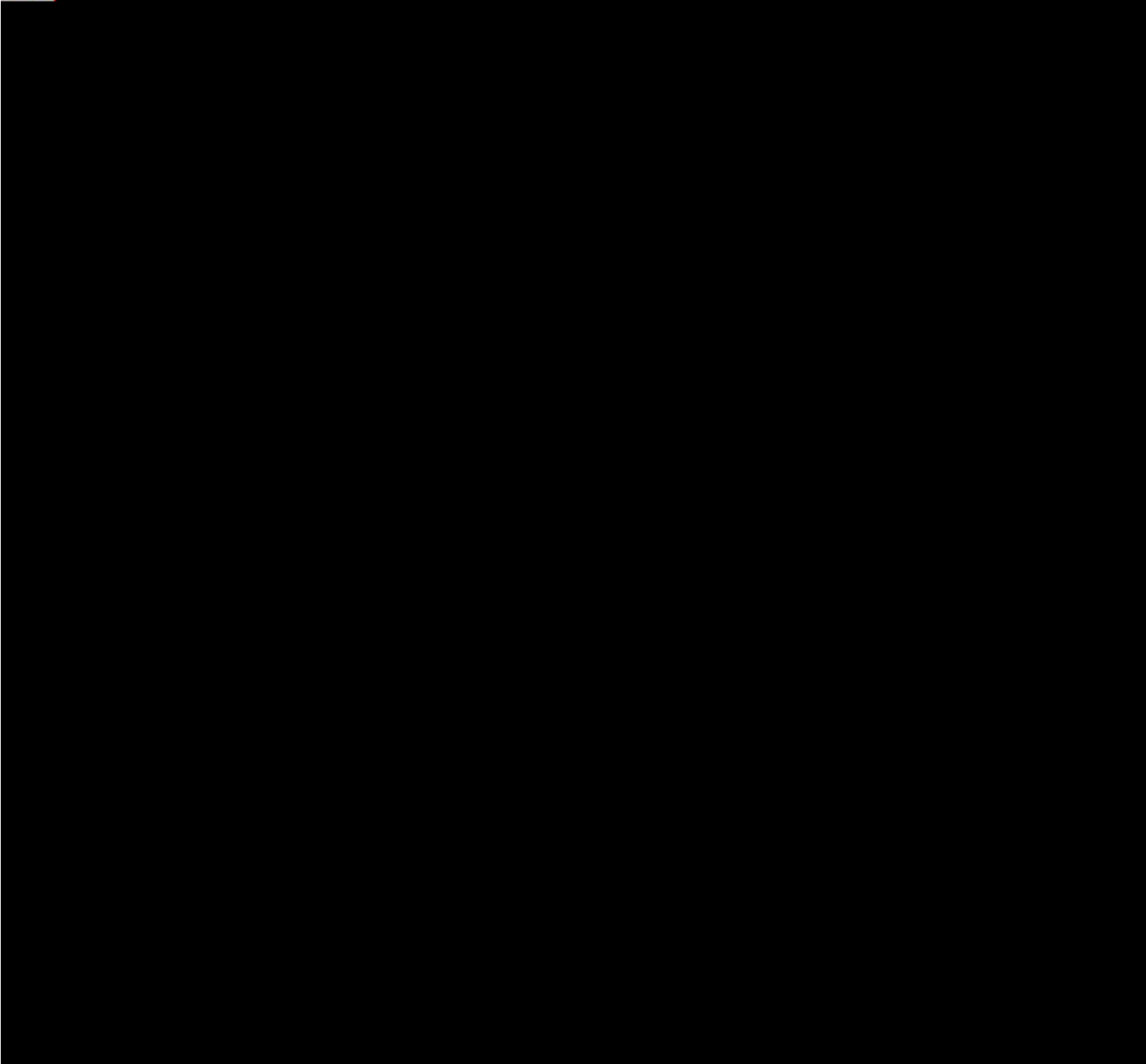


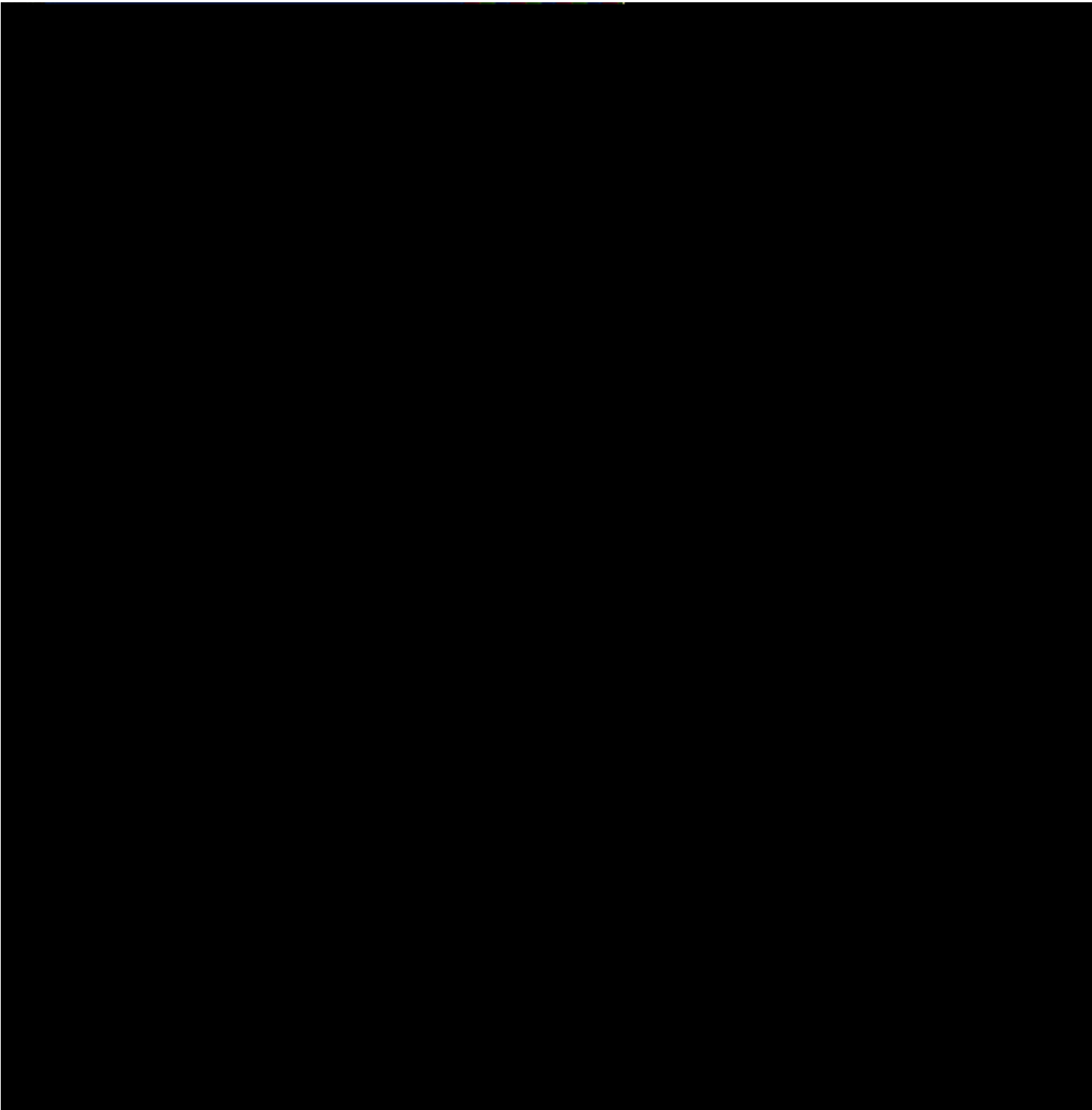
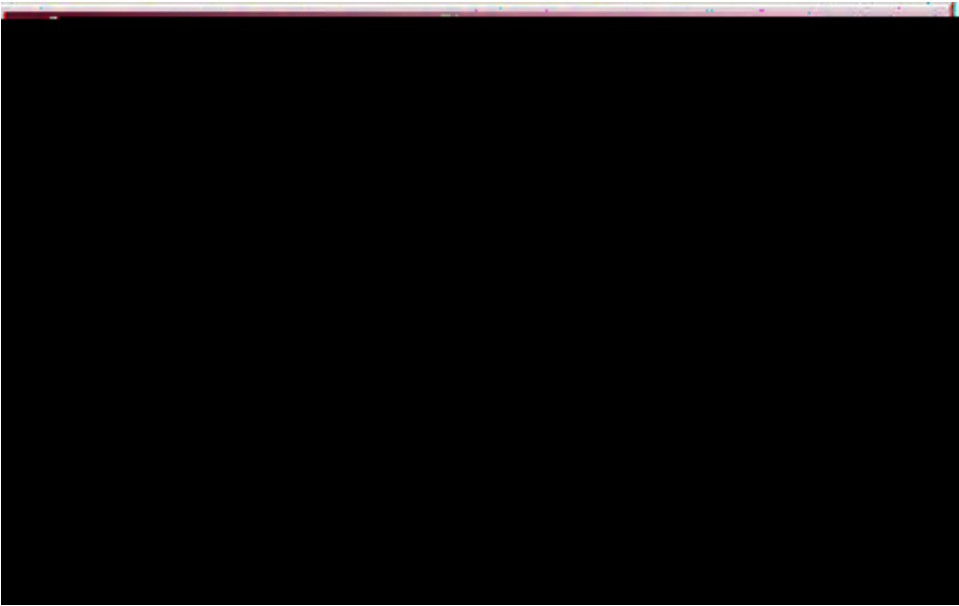
the dispatcher GUI is launched

The bottom part of the window allows the addition and removal of renderers.

2. start a user

using `c3d_add_user`





2.3. Migration of active objects

ProActive allows the transparent migration of objects between virtual machines.

A nice visual example is the [penguin's one](#).

Mobile agents

This example shows a set of [mobile agents](#) moving around while still communicating with their base and with each other. It also features the capability to move a swing window between screens while moving an agent from one JVM to the other.

1. start the penguin application

using the `penguin` script.

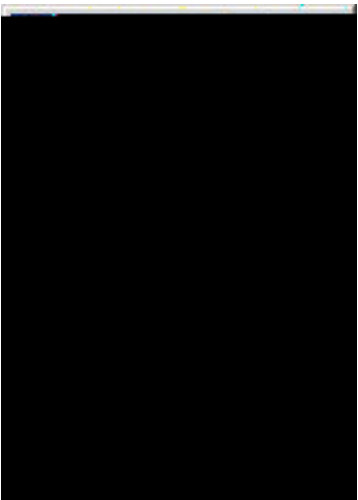
2. start IC2D to see what is going on

using the `ic2d` script

acquire the machines you have started nodes on

3. add an agent

– on the Advanced Penguin Controller window : button "add agent"



an agent is materialized by a picture in a java window.

– select it, and press button "start"

– observe that the active object is moving between the machines, and that the penguin window disappears and reappears on the screen associated with the new JVM.

4. add several agents

after selecting them, use the buttons to :

- communicate with them ("chained calls")
- start, stop, resume them
- trigger a communication between them ("call another agent")

5. move the control window to another user

– start a node on a different computer, using another screen and keyboard

– monitor the corresponding JVM with IC2D

– drag–and–drop the active object "AdvancedPenguinController" with IC2D into the newly created JVM : the control window will appear on the other computer and its user can now control the penguins application.

– still with IC2D, doing a drag–and–drop back to the original JVM, you will be able to get back the window, and control yourself the application.


```

    *
    */
public static HelloFrameController createHelloFrameController(String name) {
try {
// creates (and initialize) the active object
HelloFrameController obj =
(HelloFrameController) ProActive.newActive(HelloFrameController.class.getName(), new Object[] { name });
return obj;
} catch (ActiveObjectCreationException Ac /F3 10oudobn /F3 10p /Ftl}
}
```



```
    java.net.InetAddress localhost = java.net.InetAddress.getLocalHost();
    org.objectweb.proactive.ProActive.register(hello,
        "/" + localhost.getHostName() + "/Hello");
[ catch (Exception e) {
    System.err.println
```

```
/** method for migrating
 * @param destination_node destination node
 */
public void moveTo(String destination_node) {
    System.out.println("\n-----");
    System.out.println("starting migration to node : " + destination_node);
    System.out.println("...");
    try {
        // THIS MUST BE THE LAST CALL OF THE METHOD
        ProActive
```



