POLYMORPHISM example → oocex.c

struct base;
struct base_vtable
{   void (*dance)(struct base *);
    void (*jump)(struct base *, int how_high);
};

struct base
{   struct base_vtable *vtable;
    /* base members */
};

void base_dance(struct base *b)
{    b->vtable->dance(b);
}

void base_jump(struct base *b, int how_high)
{    b->vtable->jump(b, how_high);
}

struct derived1
{   struct base super;
    /* derived1 members */
};

void derived1_dance(struct derived1 *d)
{    /* implementation of derived1's dance function */
}

void derived1_jump(struct derived1 *d, int how_high)
{    /* implementation of derived1's jump function */
}

/* global vtable for derived1 */
struct base_vtable derived1_vtable =
{   &derived1_dance, /* you might get a warning here about incompatible pointer types */
    &derived1_jump /* you can ignore it, or perform a cast to get rid of it */
};

void derived1_init(struct derived1 *d)
{    d->super.vtable = &derived1_vtable;
    /* init base members d->super.foo */
    /* init derived1 members d->foo */
}

struct derived2
{   struct base super;
    /* derived2 members */
};

void derived2_dance(struct derived2 *d)
{    /* implementation of derived2's dance function */
}
void derived2_jump(struct derived2 *d, int how_high)
{    /* implementation of derived2’s jump function */
}

struct base_vtable derived2_vtable =
{   &derived2_dance,
    &derived2_jump
};

void derived2_init(struct derived2 *d)
{   d->super.vtable = &derived2_vtable;
    /* init base members d->super.foo */
    /* init derived1 members d->foo */
}

int main(void)
{
    /* OK! We're done with our declarations, now we can finally do some polymorphism in C */

    struct derived1 d1;
d1.init(&d1);

    struct derived2 d2;
d2.init(&d2);

    struct base *b1_ptr = (struct base *)&d1;
    struct base *b2_ptr = (struct base *)&d2;

    base_dance(b1_ptr); /* calls derived1_dance */
    base_dance(b2_ptr); /* calls derived2_dance */

    base_jump(b1_ptr, 42); /* calls derived1_jump */
    base_jump(b2_ptr, 42); /* calls derived2_jump */

    return 0;
}