Week 1

Binary classifier

$ googleimagesdownload -kf sedan_versus_pickup.txt -o sedan_versus_pickup -l 100

manual cleanup: delete some images.
peach > things with the color peach, like hammocks or iphone cases
apple > things like apple pie, cans of apple juice etc
bmw> just the logo
also generally, some 0byte files / corrupted files

Expectations

- giraffe_versus_elephant/
  - easy, very different shapes and also colors. 98%
- appel_versus_perzik/
  - tricky, very similar. perhaps 70%
- bmw_versus_mercedes/
  - may pick up on logos. 80%
- bmw_z4_versus_bmw_3/
  - easy, very different shapes. perhaps 90%
- chair_versus_table/
  - different shapes, but some odd photos. 80%
- meeuw_versus_spreeuw/
  - easy, very different shapes and also colors. 98%
- rutte_versus_harry_potter/
  - tricky, dataset is not great, for harry potter there are some group pictures. but the colors from harry are more blueisch. Also rutte is always wearing a suit. I think 80% should be feasible
- sedan_versus_pickup/
  - very different shape, should reach 85%
- south_african_giraffe_versus_masai_giraffe/
  - tricky one. I hope the NN is able to pick up on the different patterns. Maybe 60%

Uploading files

roland-macbook:fastai roland$ gcloud compute copy-files data gpu-docker-host-fromimage:~/data

WARNING: `gcloud compute copy-files` is deprecated. Please use `gcloud compute scp` instead. Note that `gcloud compute scp` does not have recursive copy on by default. To turn on recursion, use the `--recurse` flag.

Split in train/valid

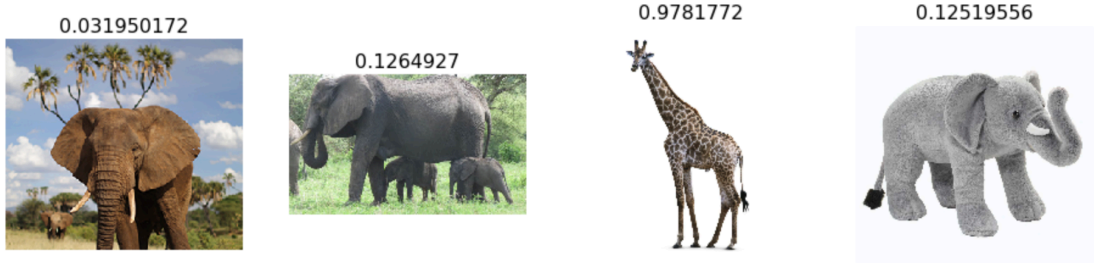
Training model
OSError: Error handling image at: /fastai/data/data/giraffe_versus_elephant/valid/elephant/56. elephant-loader.gif
I just removed this file. Perhaps gif is not a valid format.

Giraffe versus elephant

```
100%|██████████| 1/1 [00:01<00:00,  1.17s/it]
Epoch [==========] 100% 2/2 [00:02<00:00, 1.36s/it]

epoch     trn_loss   val_loss   accuracy
    0     1.210398   0.370057   0.897436
    1     0.731472   0.070873   1.0
```

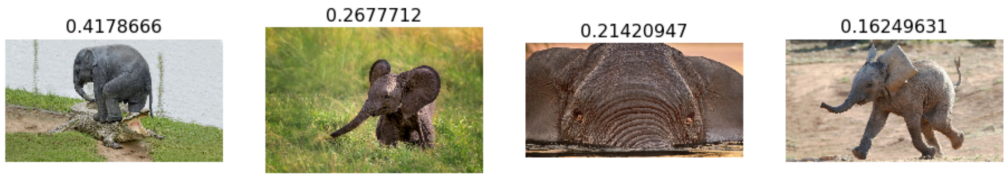
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

Correctly classified




```
In [69]: most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
         plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

Most uncertain predictions



Conclusion: my prediction of 98% accuracy was even too low! We achieved 100%.

BMW versus Mercedes

In [121]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

Epoch 100% 10/10 [00:00<00:00, 18.84it/s]

epoch	trn_loss	val_loss	accuracy
0	0.877495	1.022667	0.513514
1	0.889549	0.852195	0.540541
2	0.771642	0.741717	0.513514
3	0.67758	0.674486	0.594595
4	0.611079	0.650615	0.675676
5	0.570702	0.647646	0.72973
6	0.529626	0.669904	0.756757
7	0.476845	0.701	0.756757
8	0.440523	0.72828	0.72973
9	0.421131	0.749718	0.756757

Out[121]: [array([0.74972]), 0.7567567825317383]

Resultaat niet super. Gaat snel overfitten.

In [119]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

Correctly classified

In [120]:
```
# 2. A few incorrect labels at random
plot_val_with_title(rand_by_correct(False), "Incorrectly classified")
```

Incorrectly classified

In [124]:
```
plot_val_with_title(most_by_correct(0, True), "Most correct class 1")
```

Most correct class 1

In [125]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct class 2")
```

Most correct class 2

In [124]:
```
plot_val_with_title(most_by_correct(0, True), "Most correct class 1")
```

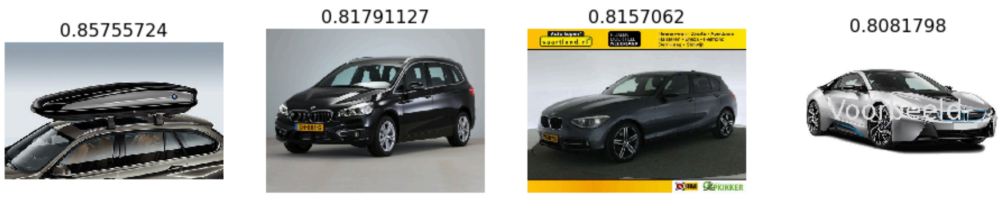
Most correct class 1

In [125]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct class 2")
```

Most correct class 2

In [126]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

Most incorrect cats

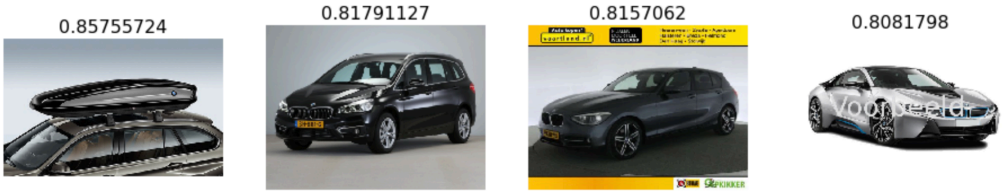


In [128]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

Most incorrect class 2

In [126]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect cats
```



In [128]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

In [129]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

Conclusion: not great results. My intuition is that we need more data.

BMW 3 series versus BMW Z4

In [160]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 3/3 [00:03<00:00,  1.02s/it]
100%|██████████| 1/1 [00:01<00:00,  1.02s/it]

Epoch 100% 10/10 [00:00<00:00, 18.56it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.872835   0.878617   0.512821
    1      0.668088   0.475407   0.74359
    2      0.53179    0.256914   0.923077
    3      0.437537   0.15862    0.948718
    4      0.365405   0.128076   0.948718
    5      0.313121   0.119094   0.948718
    6      0.268797   0.126063   0.923077
    7      0.243382   0.142961   0.923077
    8      0.216353   0.161994   0.948718
    9      0.197005   0.176308   0.948718
```

Out[160]:
```
[array([0.17631]), 0.9487179517745972]
```

In [172]:
```
plot_val_with_title(most_by_correct(0, True), "Most correct class 1")
```

```
Most correct class 1
```

In [173]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct class 2")
```

```
Most correct class 2
```

In [172]:
```
plot_val_with_title(most_by_correct(0, True), "Most correct class 1")
```

```
Most correct class 1
```

In [173]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct class 2")
```

```
Most correct class 2
```

In [174]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect class 1
<Figure size 1152x576 with 0 Axes>
```

In [175]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

In [174]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect class 1
<Figure size 1152x576 with 0 Axes>
```

In [175]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

In [176]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

Conclusion: as expected, these classes are relatively easy to distinguish.

CHAIR versus TABLE

In [198]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 1/1 [00:00<00:00,  1.06it/s]

Epoch 100% 10/10 [00:00<00:00, 20.44it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.563126   0.944147   0.529412
    1      0.524812   0.529544   0.676471
    2      0.534502   0.329214   0.852941
    3      0.414387   0.198126   0.941176
    4      0.347168   0.120709   0.970588
    5      0.292543   0.090168   1.0
    6      0.251866   0.072425   1.0
    7      0.220074   0.059451   1.0
    8      0.195416   0.055396   1.0
    9      0.199023   0.055025   0.970588
```

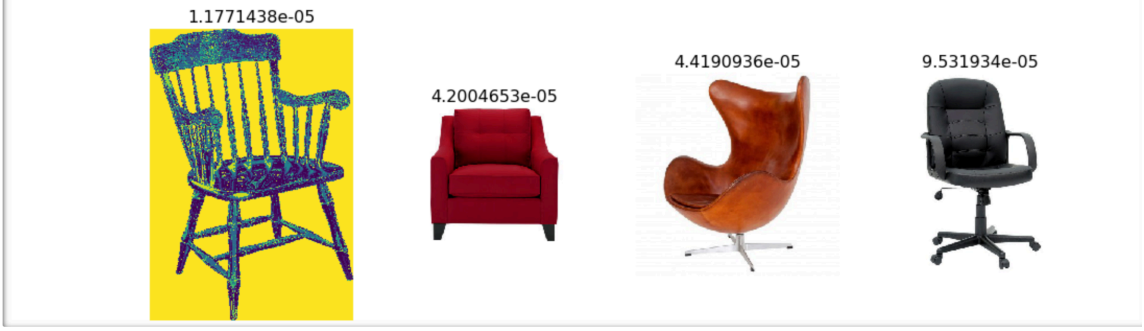
Out[198]:
```
[array([0.05503]), 0.970588207244873]
```

In [207]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```

In [212]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

```
Most correct chair
```



In [213]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```

In [215]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

In [215]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

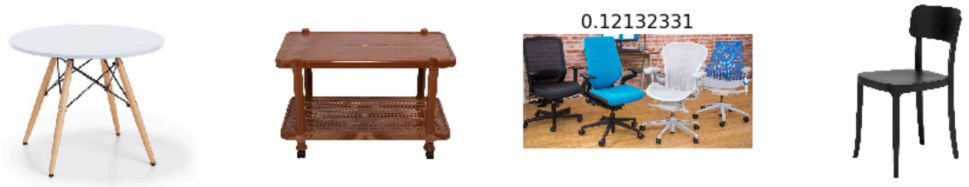
```
Most incorrect class 2
```

0.38439244

In [216]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

0.38439244 0.7602744 0.12132331 0.11358159



Conclusion: does much better than my predicted 80%! Cool.

meeuw_versus_spreeuw

In [229]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 3/3 [00:03<00:00,  1.23s/it]
100%|██████████| 1/1 [00:02<00:00,  2.32s/it]

Epoch 100% 10/10 [00:00<00:00, 17.17it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.709314   0.528505   0.74359
    1      0.548378   0.233248   1.0
    2      0.410591   0.130629   0.974359
    3      0.321861   0.09448    0.974359
    4      0.268481   0.081766   0.974359
    5      0.234602   0.085104   0.974359
    6      0.200543   0.089793   0.974359
    7      0.173074   0.095044   0.974359
    8      0.150757   0.094129   0.974359
    9      0.139575   0.093612   0.974359
```

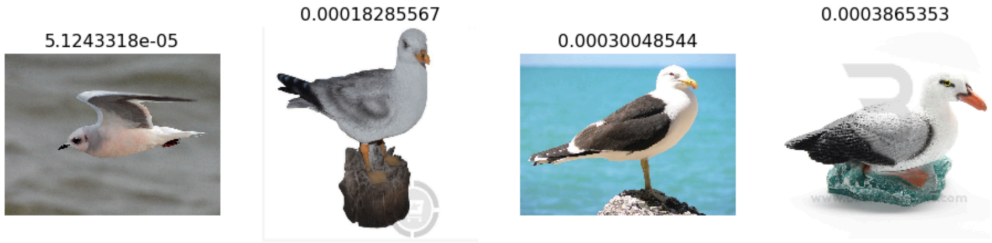
Out[229]:
```
[array([0.09361]), 0.9743589758872986]
```

In [238]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```

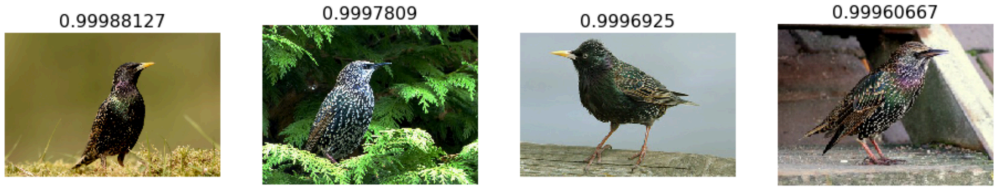
In [241]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

```
Most correct meeuw
```



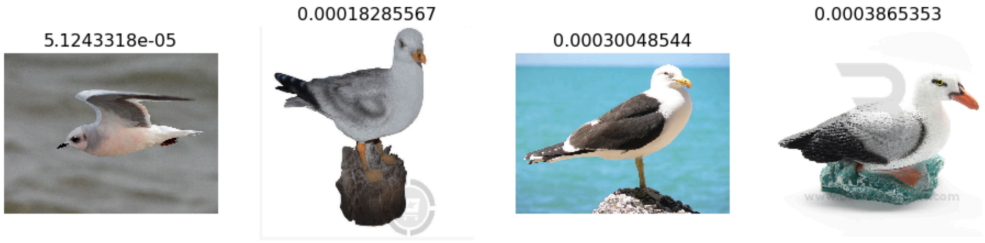
In [242]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```



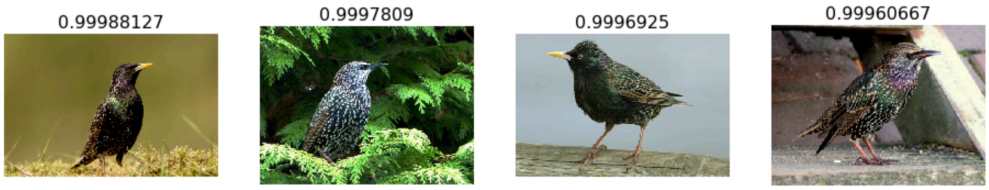
In [241]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

```
Most correct meeuw
```



In [242]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```



In [244]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

```
In [244]: plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

Conclusion: this dataset can be predicted like I expected. High accuracy given the very different classes.

SEDAN versus Pickup truck

```
In [265]: arch=resnet34
          data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
          learn = ConvLearner.pretrained(arch, data, precompute=True)
          learn.fit(0.01, 15)
```

```
100%|██████████| 3/3 [00:04<00:00,  1.39s/it]
100%|██████████| 1/1 [00:01<00:00,  1.37s/it]

      Epoch [████████████████████████████████] 100% 15/15 [00:00<00:00, 18.65it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.836447   0.466323   0.868421
    1      0.643211   0.248251   0.947368
    2      0.473987   0.175111   0.947368
    3      0.410001   0.155383   0.947368
    4      0.338402   0.14888    0.947368
    5      0.289612   0.150993   0.947368
    6      0.253184   0.151374   0.947368
    7      0.221403   0.151549   0.947368
    8      0.198052   0.152612   0.947368
    9      0.177162   0.150671   0.947368
   10      0.163218   0.148143   0.947368
   11      0.154793   0.142629   0.947368
   12      0.140006   0.140853   0.947368
```

In [278]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

```
Most correct pickup_truck
```

In [279]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```

In [278]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

```
Most correct pickup_truck
```

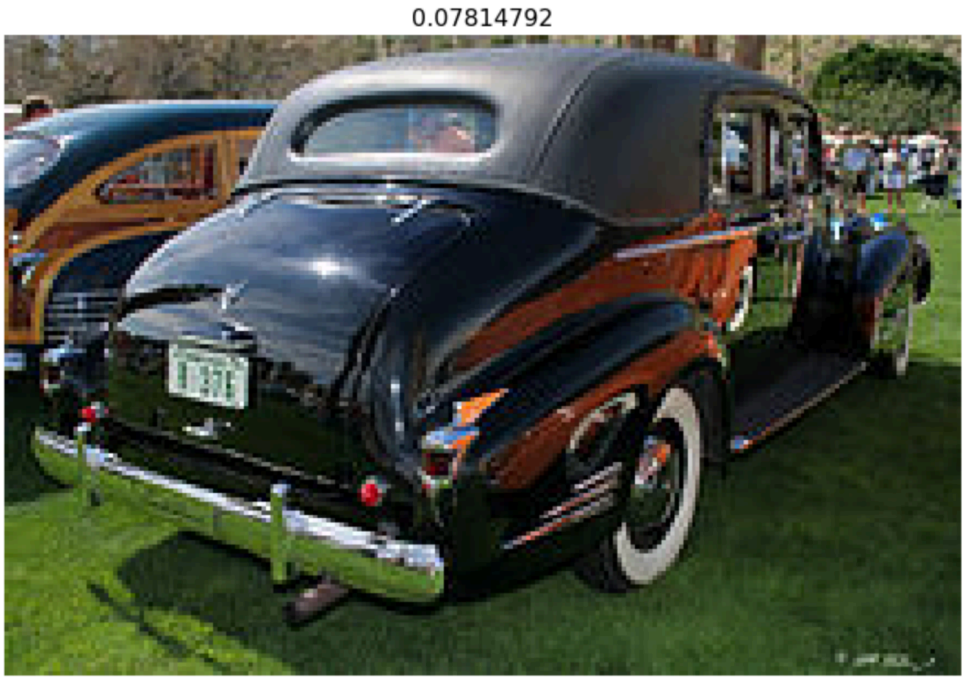
In [279]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```

In [280]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

In [281]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```



```
In [281]: plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

I think the “most incorrect” are actually correct! its just that the labels are wrong!

```
In [282]: most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

These are fairly small, vague pictures.

Conclusion: 85% was easily achieved.

Rutte versus Harry Potter

In [295]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 3/3 [00:02<00:00,  1.03it/s]
100%|██████████| 1/1 [00:00<00:00,  1.29it/s]

Epoch 100% 10/10 [00:00<00:00, 19.27it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.710575   0.479968   0.8
    1      0.510306   0.270477   0.942857
    2      0.392461   0.146947   1.0
    3      0.310633   0.117591   0.942857
    4      0.261007   0.098725   0.971429
    5      0.363741   0.106628   0.971429
    6      0.308863   0.082998   0.971429
    7      0.307874   0.061038   0.971429
    8      0.391125   0.076511   0.971429
    9      0.346723   0.100858   0.942857
```

Out[295]:
```
[array([0.10086]), 0.9428571462631226]
```

In [304]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```

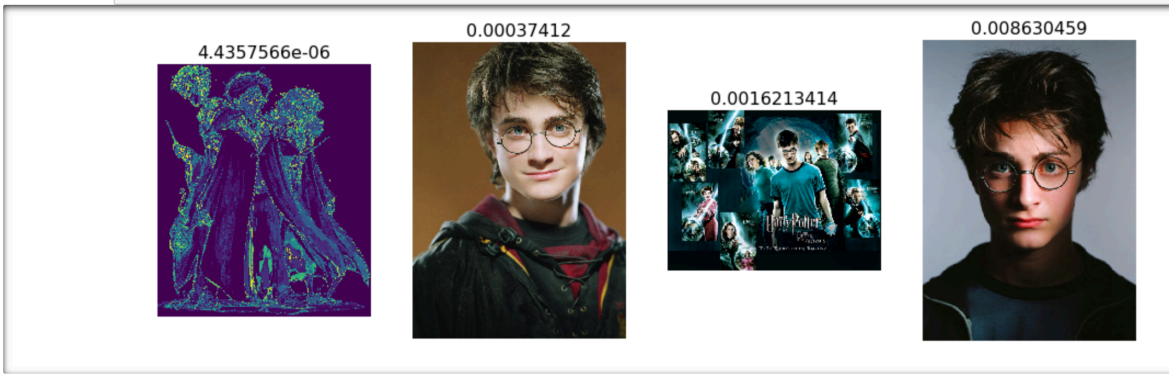
In [304]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```

In [310]:
```
# 2. A few incorrect labels at random
plot_val_with_title(rand_by_correct(False), "Incorrectly classified")
```

```
Incorrectly classified
```

In [311]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```



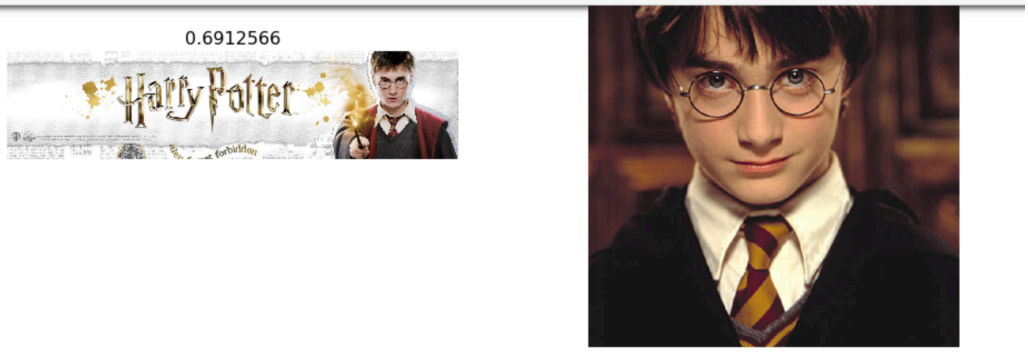
In [311]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

In [311]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

In [312]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```

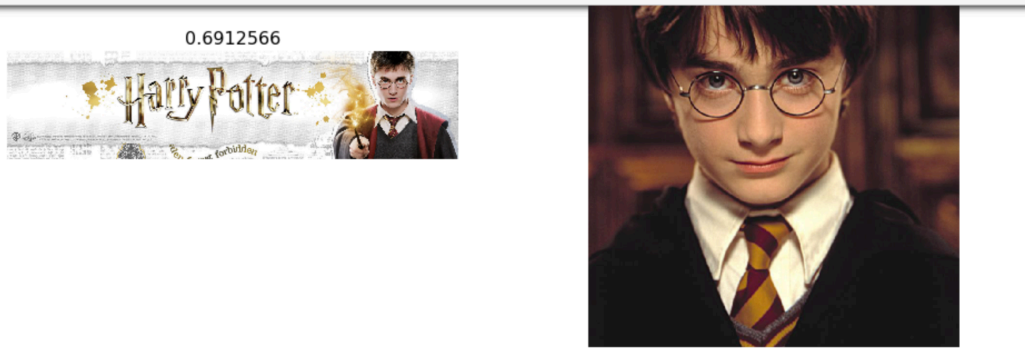
In [313]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```



In [314]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
<Figure size 1152x576 with 0 Axes>
```

In [313]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```



In [314]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
<Figure size 1152x576 with 0 Axes>
```

In [315]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

In [315]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

Conclusion: again, the NN performed much better than I expected. Though I am not sure whether it can really recognize these 2 persons. I think the style of the photos is just really different. Mental note: do a comparison of various politicians or actors.

Last: South African Giraffe versus Masai Giraffe

In [327]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 3/3 [00:05<00:00,  1.70s/it]
100%|██████████| 1/1 [00:01<00:00,  1.96s/it]

      Epoch [████████████████████████████████] 100% 10/10 [00:00<00:00, 17.28it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.755909   0.806627   0.45
    1      0.729105   0.759772   0.5
    2      0.684765   0.71894    0.575
    3      0.633249   0.695716   0.55
    4      0.580498   0.718255   0.575
    5      0.51721    0.747415   0.55
    6      0.478577   0.779054   0.575
    7      0.447871   0.822055   0.575
    8      0.409474   0.847211   0.575
    9      0.376428   0.862352   0.575
```

Out[327]:
```
[array([0.86235]), 0.574999988079071]
```

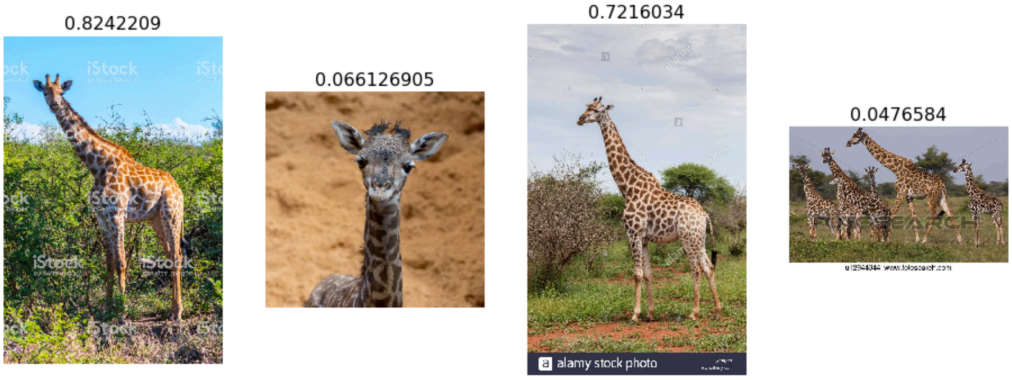
In [327]:
```
arch=resnet34
data = ImageClassifierData.from_paths(PATH, tfms=tfms_from_model(arch, sz))
learn = ConvLearner.pretrained(arch, data, precompute=True)
learn.fit(0.01, 10)
```

```
100%|██████████| 3/3 [00:05<00:00,  1.70s/it]
100%|██████████| 1/1 [00:01<00:00,  1.96s/it]

Epoch 100% 10/10 [00:00<00:00, 17.28it/s]

epoch      trn_loss   val_loss   accuracy
    0      0.755909   0.806627   0.45
    1      0.729105   0.759772   0.5
    2      0.684765   0.71894    0.575
    3      0.633249   0.695716   0.55
    4      0.580498   0.718255   0.575
    5      0.51721    0.747415   0.55
    6      0.478577   0.779054   0.575
    7      0.447871   0.822055   0.575
    8      0.409474   0.847211   0.575
    9      0.376428   0.862352   0.575
```

Out[327]:
```
[array([0.86235]), 0.574999988079071]
```

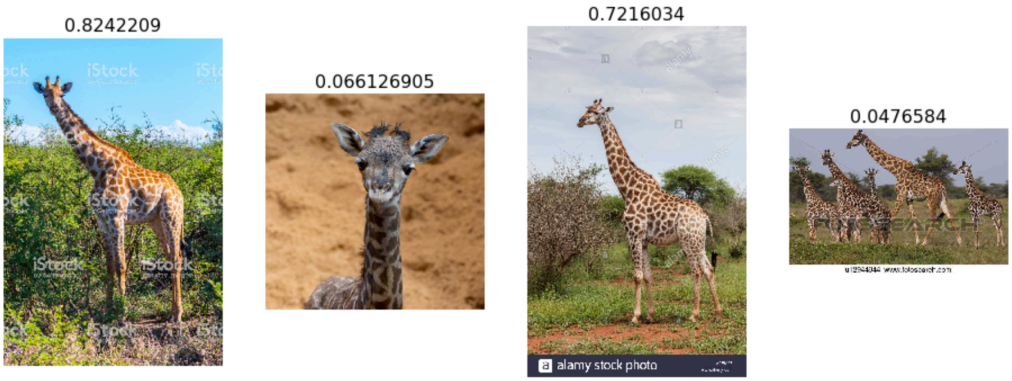
In [336]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```



In [336]:
```
# 1. A few correct labels at random
plot_val_with_title(rand_by_correct(True), "Correctly classified")
```

```
Correctly classified
```



In [337]:
```
# 2. A few incorrect labels at random
plot_val_with_title(rand_by_correct(False), "Incorrectly classified")
```

```
Incorrectly classified
```

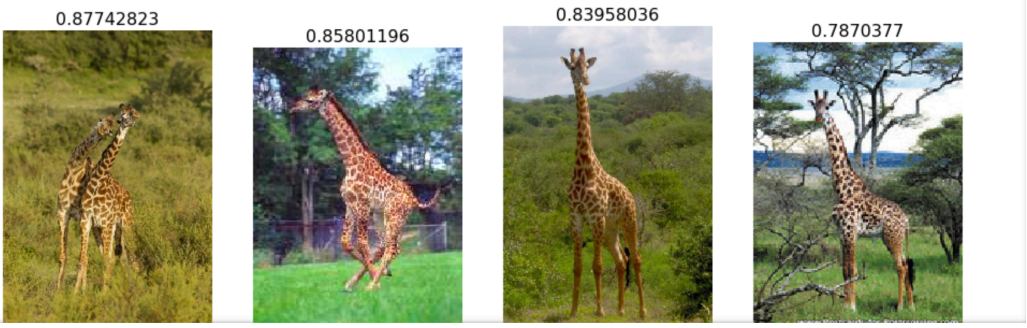
In [339]:
```
plot_val_with_title(most_by_correct(0, True), f"Most correct {class1}")
```

In [340]:
```
plot_val_with_title(most_by_correct(1, True), "Most correct {class2}")
```

```
Most correct {class2}
```

In [341]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect class 1
```

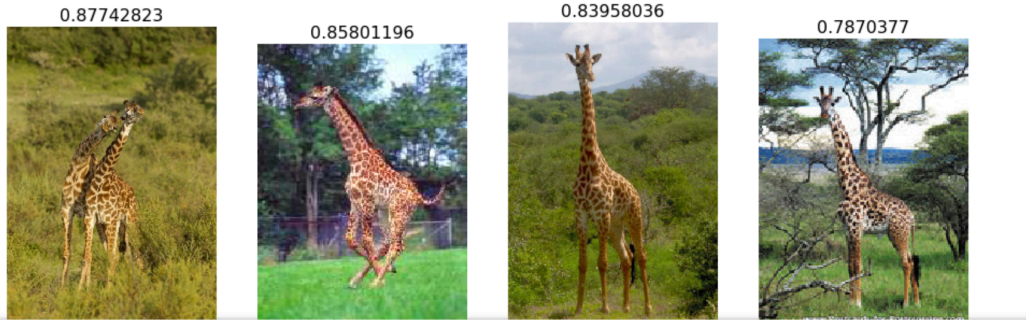


In [342]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

In [341]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect class 1
```

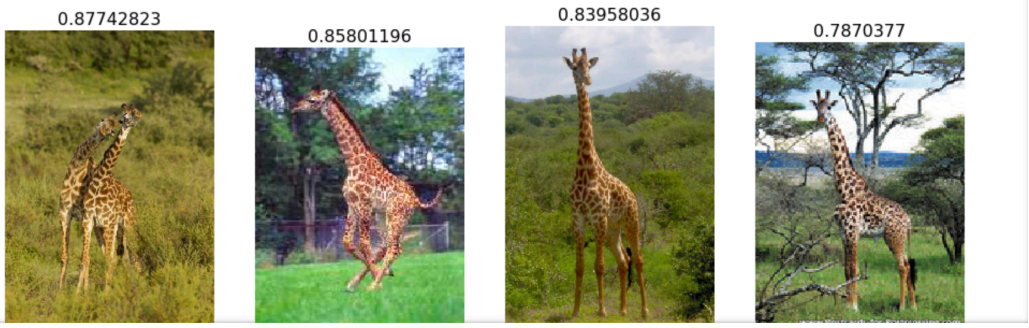


In [342]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```

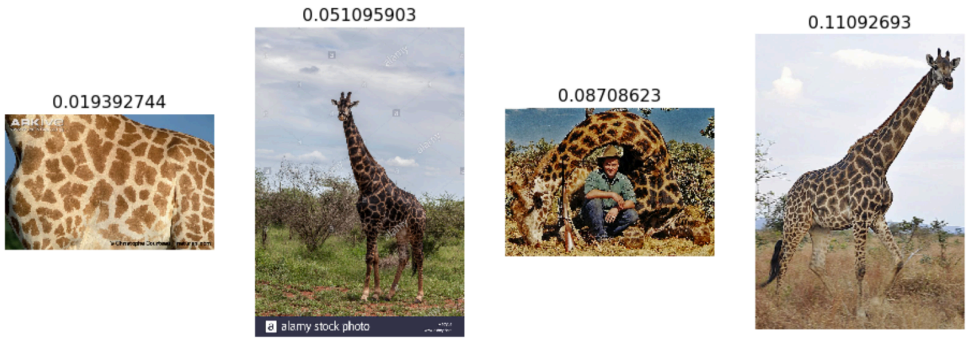
In [341]:
```
plot_val_with_title(most_by_correct(0, False), "Most incorrect class 1")
```

```
Most incorrect class 1
```



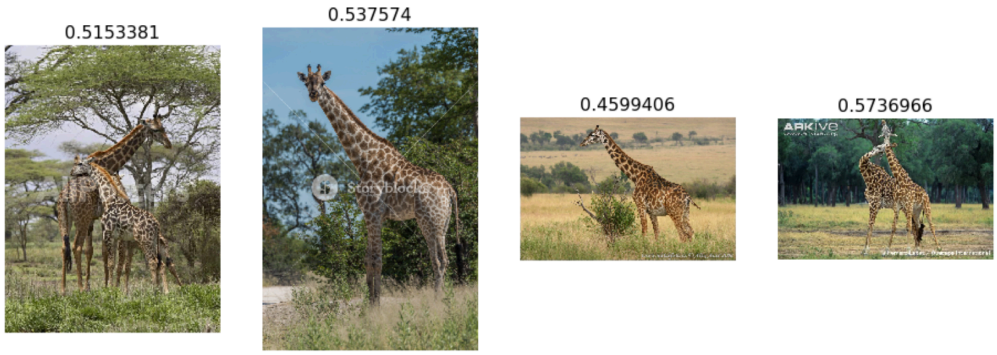
In [342]:
```
plot_val_with_title(most_by_correct(1, False), "Most incorrect class 2")
```

```
Most incorrect class 2
```



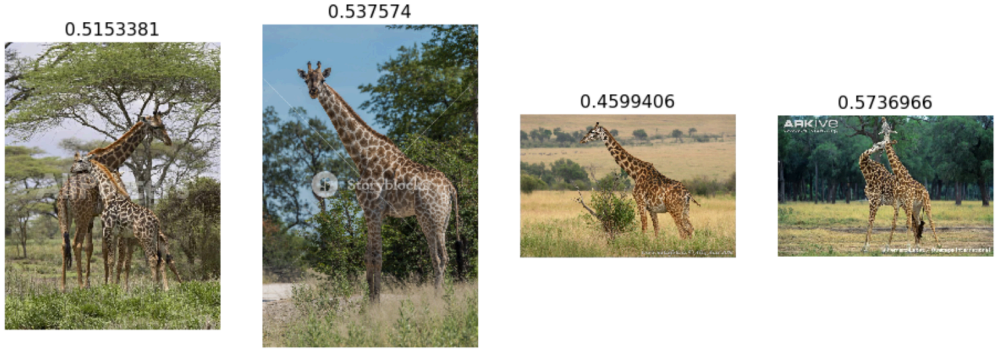
In [343]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```



In [343]:
```
most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```



Conclusion: as I expected, a rather hard task. The NN does slightly better than chance. Worse than I predicted. This is an interesting challenge to pick up next. I feel that some tuning of the model, or more data / data augmentation can really help here.

TODO apple versus peach